让你变懒的 Android Studio Live Templates

原创 2016-08-22 stormzhang AndroidDeveloper

阅读本文大概需要 6.66 分钟。

俗话说，不想偷懒的程序员不是好程序员！那么今天就教大家偷懒下！

先举个例子，我们在 Android 开发中输入 *Toast* ，然后会快速弹出如下完整代码：

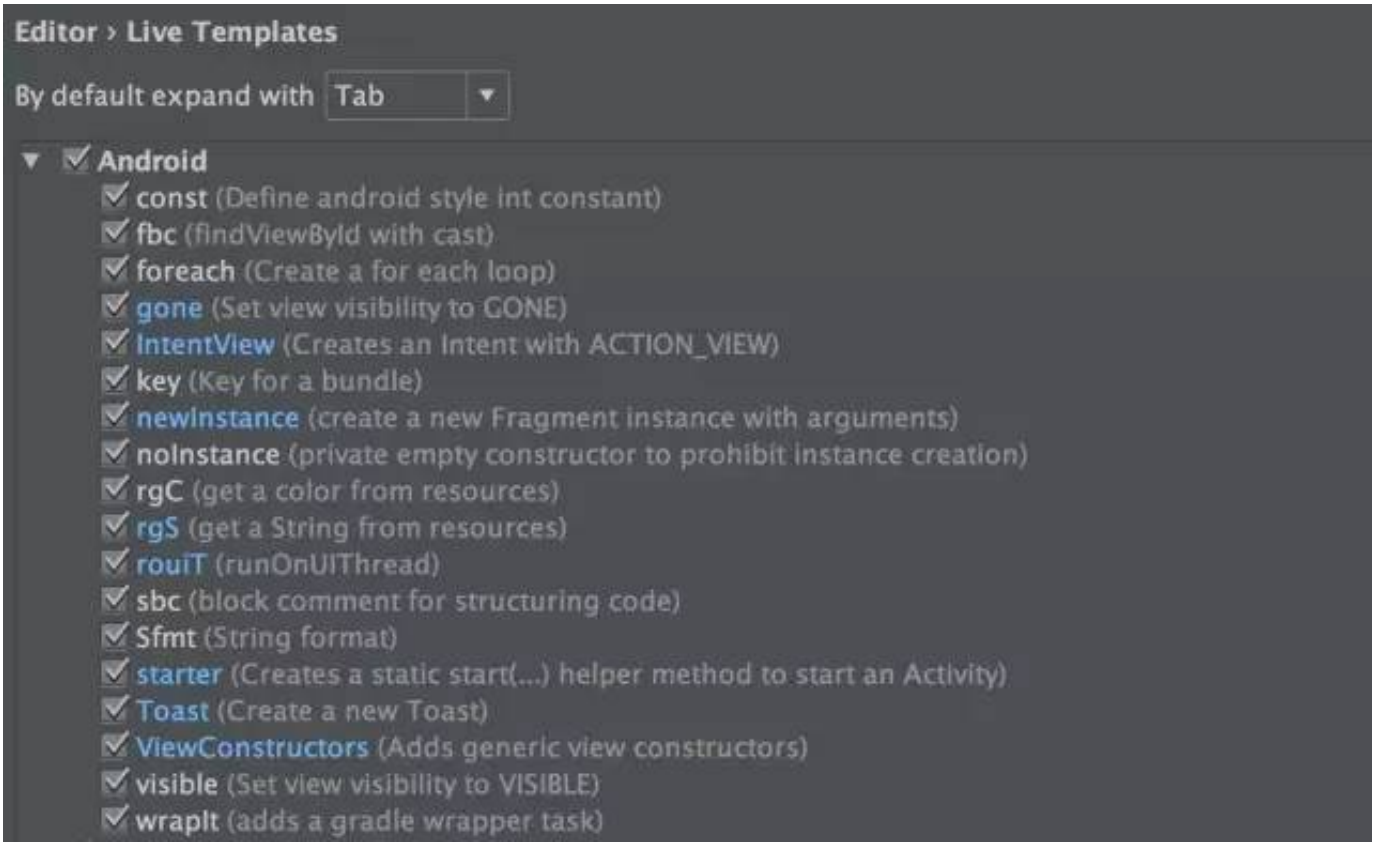
Toast.makeText(MainActivity.this, "", Toast.LENGTH_SHORT).show();

然后直接输入 text 内容就可以了，是不是很方便?

有同学问，这不就是自动补全么？错了，乍一看是自动补全，其实不然。自动补全是对一个方法或类名的补全，比如你输入 *find* 然后就会有 *findViewById* 方法提示你，你可以快速定位，但是我们实际开发中一般还需要对它强制转型，然后加上 *R.id.xx* 来声明它的 id，但是你可以试着输入 *fbc* ，然后按 *tab* 或者 *enter* 键，你会发现比自动补全还要更方便。

这个就叫做 *Live Template* ，如果真要翻译姑且就叫做实时模板吧，在 AS 中有两种模板，一种就是你在新建一个 Activity 的时候可以选择 Empty Activity、FullScreen Activity 之类的，这个一般是对你整个文件而言的，还有一种就是本篇要介绍的 *Live Template* ，这个会在一些常用到的代码片段会非常有用。

打开 *设置 -> Editor -> Live Templates* ，可以看到默认已经有很多 *Live Templates* 了，可以看下我的截图 Android 分类下有如下这些模板:



你都可以输入那些缩写来快速启用这些模板，比如定义一个常量，快速设置 View 为 gone ，快速启动一个 Activity 等，你都可以直接输入 *const*、*gone*、*starter* 来快速操作，是不是觉得很酷炫！

那具体是如何实现的呢？我们不妨点击这些模板列表的 *starter* ，有如下代码：

```
public static void start(Context context) {
    Intent starter = new Intent(context, $ACTIVITY$.class);
    starter.putExtra($CURSOR$);
    context.startActivity(starter);
}
```

其中 *$ACTIVITY$* 代表当前所在的类名，*$CURSOR$* 代表当前鼠标的定位位置，同理 *newInstance* 可以帮你在 Fragment 中快速声明一个新建 Fragment 的方法，它的代码如下：

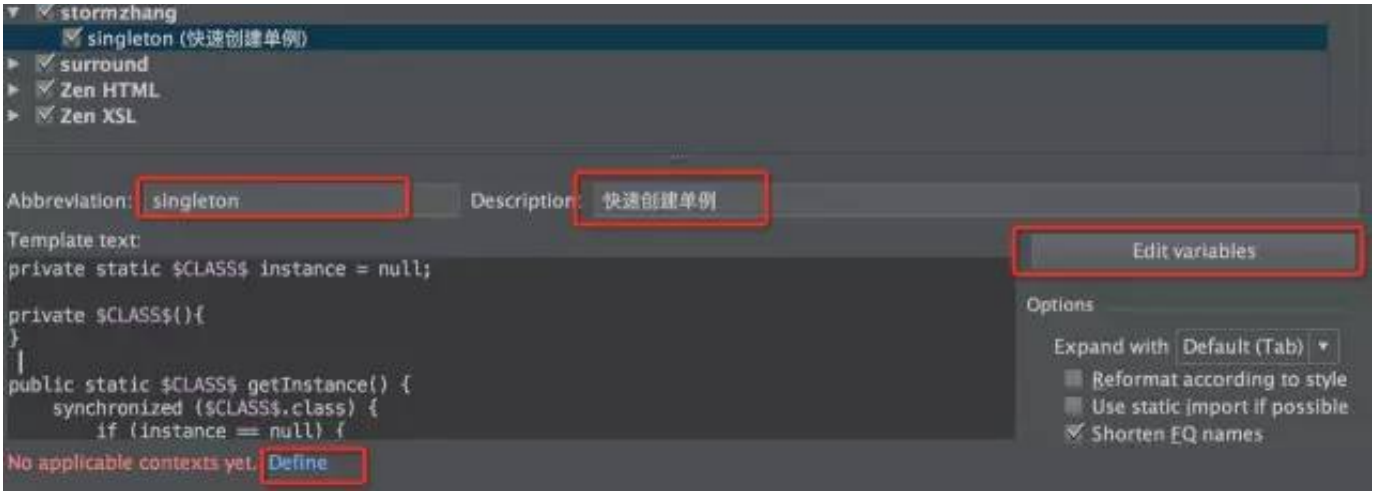
```
public static $fragment$ newInstance($args$) {
    $nullChecks$
    Bundle args = new Bundle();
    $addArgs$
    $fragment$ fragment = new $fragment$();
    fragment.setArguments(args);
    return fragment;
}
```

其中 *$$* 代表是一个变量，中间包裹着这个变量的名字，你可以对这个变量声明类型，这个后面再说。

是不是很容易理解呢？如果理解了那么就可以来根据自己的使用习惯来定义自己的 *Live Templates* 了。

比如我们在开发中要经常写单例模式吧？每次都要写这么一大段是不是很烦？那么今天就教大家自定义一个单例模式的模板，以后轻松搞定单例。

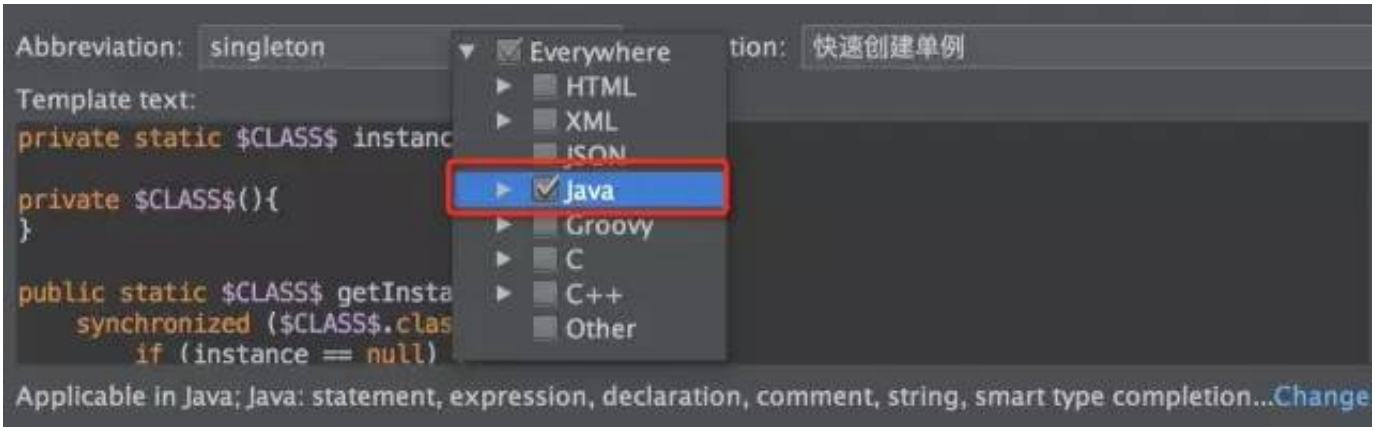
到 *设置 -> Editor -> Live Templates* ，点击右上角的 + 号，选择 *Template Group* ，因为我习惯自定义的单独分组先，这样好管理，比如新建一个分组叫做 *stormzhang* ，然后就会看到有一个 *stormzhang* 的分组显示在了列表里，这时候鼠标选中该分组，然后再点击右上角的 + 号，点击 *Live Template* ，然后如下图填写缩写与描述：



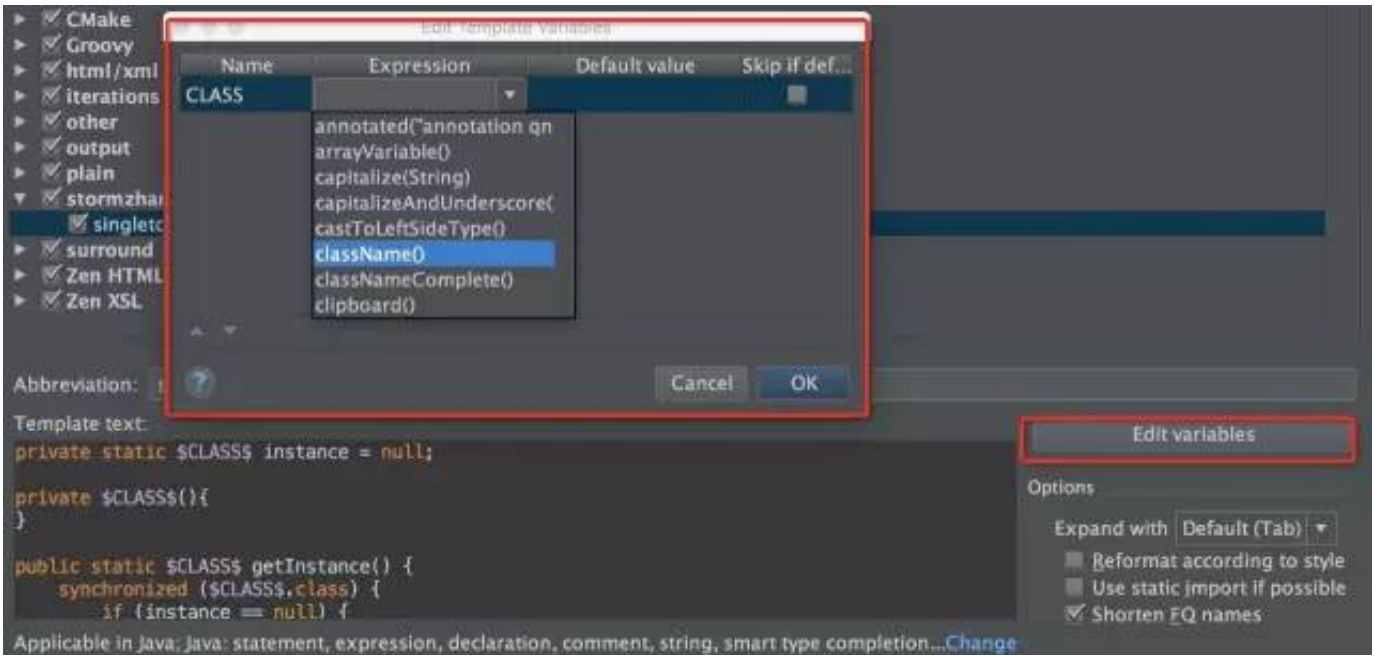
紧接着把如下代码拷贝到下面的输入框里（PS：单例模式的写法有很多种，这里就随意以其中一种为例）

```
private static $CLASS$ instance = null;

private $CLASS$(){
}

public static $CLASS$ getInstance() {
    synchronized ($CLASS$.class) {
        if (instance == null) {
            instance = new $CLASS$();
        }
    }
    return instance;
}
```

注意这里，如果你这段代码是一些固定的代码，那么至此就结束了，但是这段代码里是动态的，里面有一些变量，因为每个类的类名如果都需要自己手动更改就太麻烦了，所以有个变量 *$CLASS$* ，所以需要点击下面的 *Define* ，先要定义变量所属的语言范围，点开之后可以看到这里支持 HTML、XML、JSON、Java、C++ 等，很明显，我们这里需要支持 Java ，这里选中 Java :



紧接着，我们需要给变量 *$CLASS$* 定义类型，这里的 *CLASS* 名字随意取的，为了可读性而已，你高兴可以取名 *abc* ，真正给这个变量定义类型的是点击 *Edit variables* 按钮，来对该变量进行编辑，我们选择 *calssName()* 选项，可以看到还有其他选项，但是看名字大家大概就猜到什么含义了，这里就不一一解释了。



点击 *ok* 保存，至此我们定义的一个单例的 Live Template 就完成了。你可以随意打开一个类文件，然后输入 *singleton* 按 *tab* 或者 *enter* 键就可以看到神奇的一幕出现了，是不是很帅？

看完这篇文章想想自己还有哪些常用到的代码片段，赶紧把它定义成一个 Live Template 吧，你会发现你又可以变懒了！

PS：被小人投诉，我的赞赏功能终于回归了，从此又可以任性的装逼了！

阅读原文