# 官网完成快捷键：

https://developer.android.com/studio/intro/keyboard-shortcuts.html，在“09\_技术文章收藏”中下载了离线网页“键盘快捷键 \_ Android Studio.html”

# 代码自动完成

关于代码自动完成： https://www.jetbrains.com/help/idea/2016.1/auto-completing-code.html、https://www.jetbrains.com/help/idea/2016.1/code-completion-2.html

Android Studio 有三种代码自动完成类型，您可以使用键盘快捷键访问它们。

**表 2.** 代码自动完成的键盘快捷键。

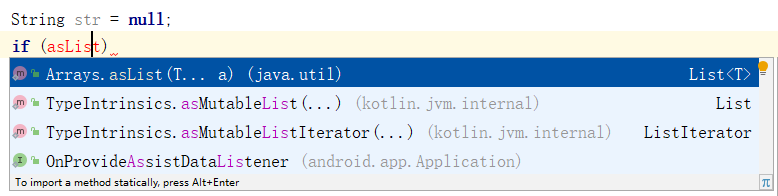
|  |  |  |  |
| --- | --- | --- | --- |
| 类型 | 说明 | Windows 和 Linux | Mac |
| 基本自动完成 | 显示对变量、类型、方法和表达式等的基本建议。 如果连续两次调用基本自动完成，将显示更多结果，包括私有成员和非导入静态成员。 | **Control+空格** | **Control+空格** |
| 智能自动完成 | 根据上下文显示相关选项。 智能自动完成可识别预期类型和数据流。 如果连续两次调用智能自动完成，将显示更多结果，包括链。 | **Control+Shift+空格** | **Control+Shift+空格** |
| 语句自动完成 | 为您自动完成当前语句，添加缺失的圆括号、大括号、花括号和格式化等。 | **Control+Shift+Enter** | **Shift+Command+Enter** |

您还可以按 **Alt+Enter** 执行快速修复并显示建议的操作。

Ctrl + 空格是控制输入法的，所以一般把这个改成Ctrl + 1

基本自动完成：

比如我记得有个把数组转换为列表的方法，只记得方法名，不记得是哪个类中的方法了，这里可以先输入方法名，然后按两次Ctrl + 1，如下：

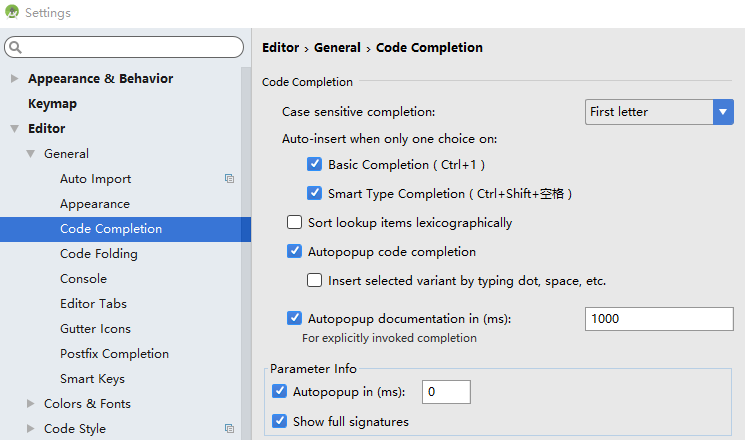


按回车完成输入。 按Alt + Enter，可以选择静态导入这个方法，如下：



代码完成设置：

File-->Settings-->Edittor-->Genoral-->Code Completion

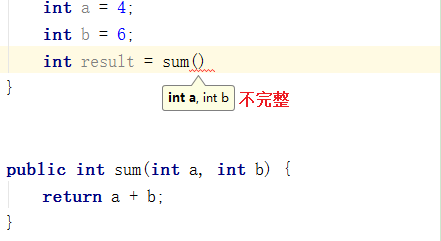
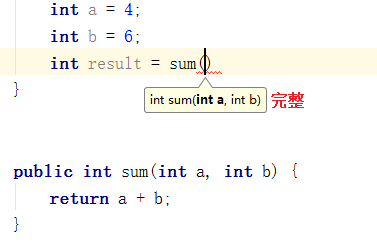


Autopopup code completion(自动弹出代码完成），在输入的时候就自动弹出代码完成列表，如果列表消失了，可以按‘Ctrl + 1’或‘Ctrl + Shift + 空格’来再次显示。

Patametor Info (参数信息）

Autopopup in (ms): 在多少毫秒内自动弹出参数信息，默认是1000，建议修改为0（立即弹出）

Show full signatures 显示完整签名（建议勾选，可以看到返回类型），不完整和完整的区别如下：

在参数隐藏后可按Ctrl + P再次显示

# 快捷键

## 常用

Tab缩进的反操作：Shift + Tab

光标定位到类名上，按Shift + F1，在浏览器中打开doc文档

按信Ctrl再把光标移动变量名或者类名方法名等上面，会有一个简单的信息，如看到变量名的声明类型等

选择一些代码，按Ctrl + Alt + T，出现选择包围块，如if、while、try / catch等

Ctrl + V 粘贴最近的一个复制，Ctrl + Shift + V，从多个最近的复制中选择以一个进行粘贴操作

Ctrl + Shift + F 在指定的位置进行查找

Ctrl + Shift + R 在指定的位置进行查找替换

Ctrl + Shift + F7 高亮显示光标所在方法/字段在文件中的使用

Ctrl + Shift + F7 跳转到光标方法/字段的调用处，如果有多处调用弹框选择

Alt + F7 查找光标所在方法/字段的调用

Ctrl + Shift + J 把多行代码放到一行显示

Ctrl + J 插入模板

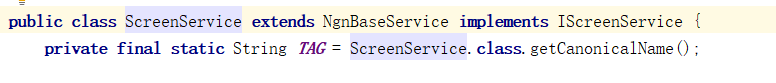
Alt + 上下箭头，快速在方法之间进行定位

F2/ Shift + F2 跳转到下一个/前一个 高亮的错误

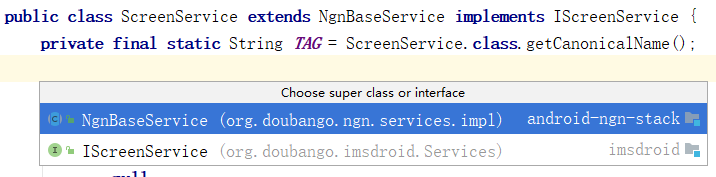
跳转到一个实现类：



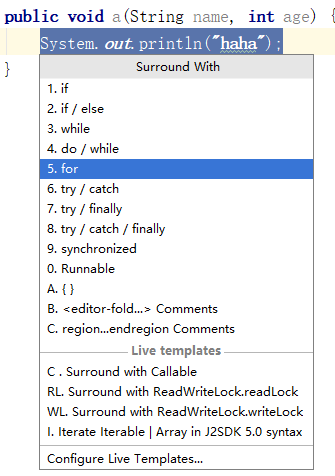
把光标定位到IScreenService，然后按Ctrl + Alt + B，效果如下：



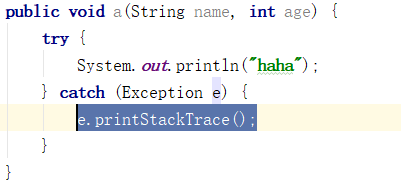
跳转到父类，在当前类中按下Ctrl + U，跳到父类，如果有多个父类，会弹出选择框：



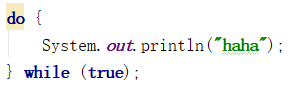
快速使用语句块包围你选择的代码块，Ctrl + Alt + T，如要把一些代码try cacth，先选择代码，再按快捷键，如下：



如果选择了try/ catch，效果如下：



如果选择了do while，如下：



当光标不在代码区时，按下Esc则可使光标定位到代码编辑器

单行注释：Ctrl + /

多行注释：Ctrl + Shift + /

搜索指定路径中的内容：Ctrl + Shift + F

替换指定路径中的内容：Ctrl + Shift + R

自动缩进行（或多行）：Ctrl + Alt + I

重新格式化代码：Ctrl Alt + L

上、下移动代码：Ctrl + Shift + ↑↓ Alt + Shift + ↑↓

滚动滚动条：Ctrl + ↑↓

窗口中打开了多个文件，左右切换打开的文件：Alt + 左右箭头

双击单词可选择整个单词，在一行中三击，则选中整行。

结构化选择：将光标置于任意单词，按下Ctrl + W，选择整个单词，继续按，则选中的区域会扩展至包含任意数量的相邻文本，按Ctrl + Shift + W，则选中区域会缩小

F2：跳到报错的地方

抽取方法：Ctrl + Alt + M

抽取变成员：Ctrl + Alt + F

抽取局部变量：Ctrl + Alt + V

抽取常量：Ctrl + Alt + C

大小写转换：Ctrl + Shift + U

复制文件：F5

重命名：Shift + F6

实现抽象方法：Ctrl + I (Implements)

覆盖方法：Ctrl + O(Override)

查看方法参数提示：Ctrl + P

查看文档说明：Ctrl + Q

查看方法代码：Ctrl + Shift + I

查看类继承关系：Ctrl + H

查看某个变量、方法被调用：alt + F7

查看方法的层次调用：Ctrl + Alt + H (Hierarchy Callers of requestData)

查看子类：Ctrl + Alt + B

高亮选择的代码块：Ctrl + Shift + F7 取消高亮：按两下Esc

查看当前Activity对应的布局，或者当前布局对应的Activity：Ctrl + Alt + Home

折叠代码：

* Ctrl + +/-，折叠光标当前
* Ctrl + Shift + +/-，折叠整个类

**代码提示：Ctrl + Shift + 空格**

在当前光标下面插入一行空行：Sift + Enter

在当前光标上面插入一行空行：Ctrl + Alt + Enter

工具窗口快捷键Alt + 1234567，按一下打开，再按一下关闭

隐藏/显示工具窗口（最大化/最小化编辑器） Ctrl + Shift + F12

导航：

导航到类文件：Ctrl + N

导航到文件：Ctrl + Shift + N，也可导航到文件夹，如果只导航到文件夹，可以在末尾加'/'符号

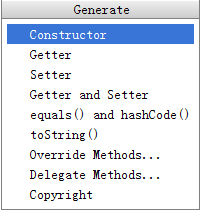
导航到符号：Ctrl + Shift + Alt + N

Ctrl + F12，快速定位方法或者变量，在输入方法时可以使用单词首字母，如oCr查找到onCreate方法

打开资源文件：双击Shift

在布局预览中，选择控件按下Ctrl + B进入到该控件的xml位置

Fn + Alt + Insert

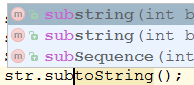


在编辑器选项卡之间导航：Alt + 左右箭头，关于编辑器选项卡的命令在：Window🡪Editor Tabs，在编辑器选项卡上右击可出现相关的上下文菜单。

语句补全：Ctrl + Shift + Enter，如输入if后按这个快捷键，得到如下：

**if** () {  
}

代码补全：Enter和Tab区别：

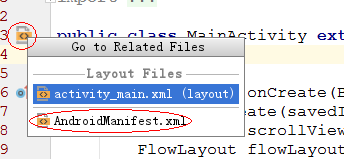


 按Enter的效果

按Tab的效果，直接把之前的toString给覆盖了

全屏代码窗口：Shift + F12

在Activity中打开清单文件：

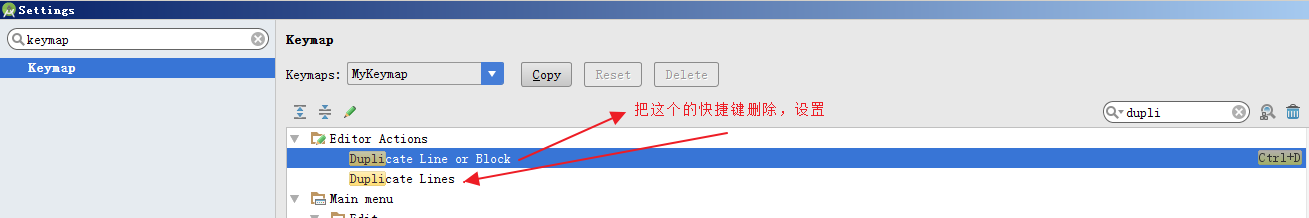


打开最近编辑过的文件：Ctrl + Shift + E （Recently Edited Files）

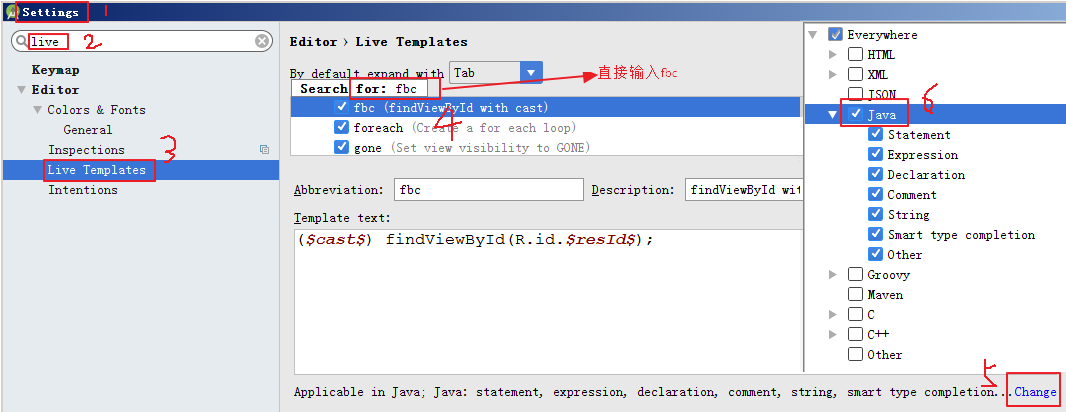
打开最近访问过的文件：Ctrl + E （Recent Files）

把Basic快捷键改为Ctrl + 1

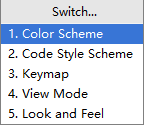
修改Ctrl + D 快捷键



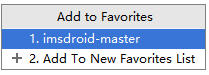
findViewById快捷输入

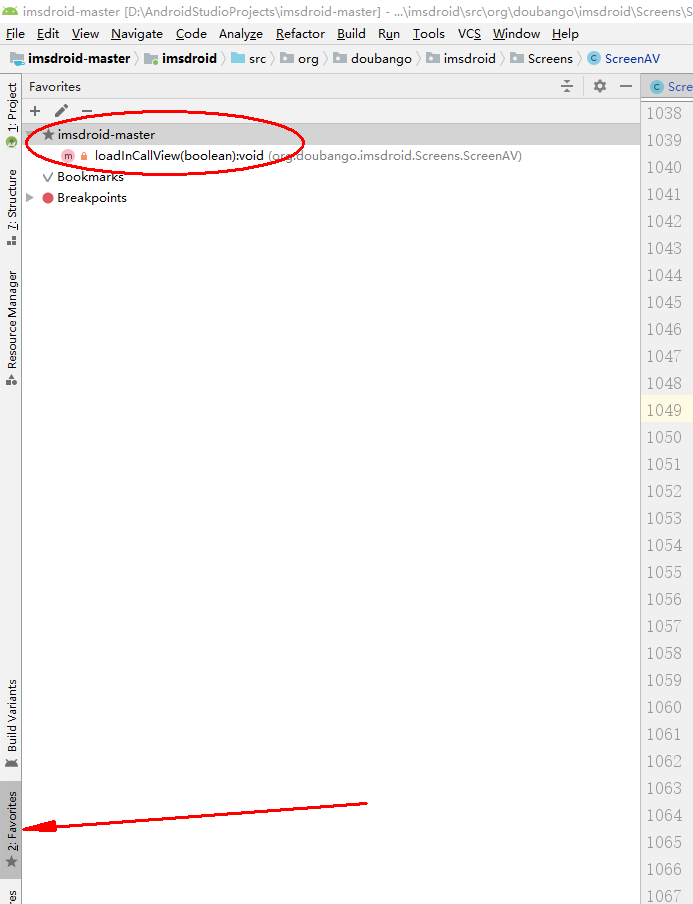


快速切换模式 Control + `（反引号）

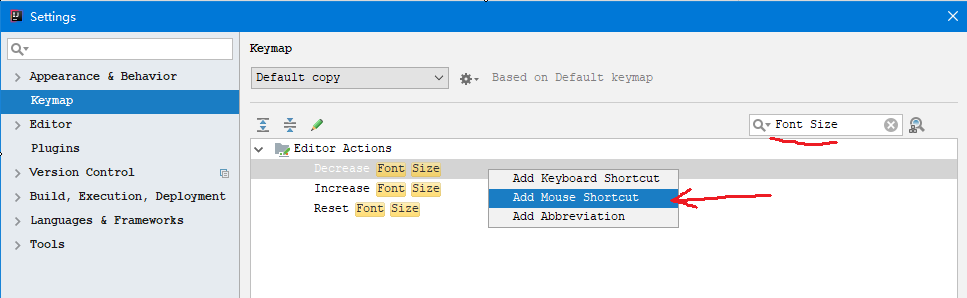


收藏一些代码位置到“收藏夹” Alt + Shift + F





## 设置字体放大缩小

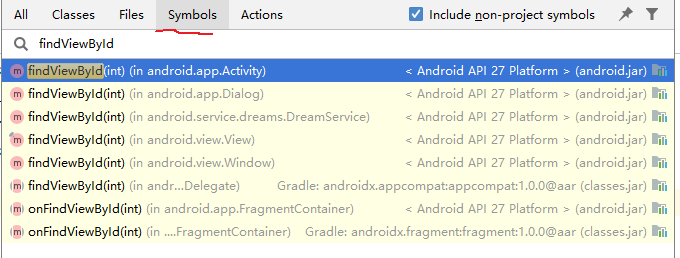


Decrease为缩小，Increase为放大，把快捷键设置Mouse Shortcut，即鼠标方式，设为Ctrl + 滚轮的上下滑动为放大和缩小

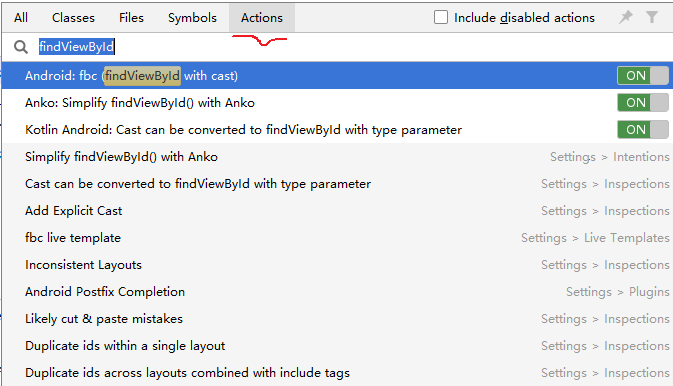
## 双击Shift

All可搜索类、文件、符号、动作

符号可以搜索方法、文件名等，如下：



Action可以搜索快捷键，设置等，如下：

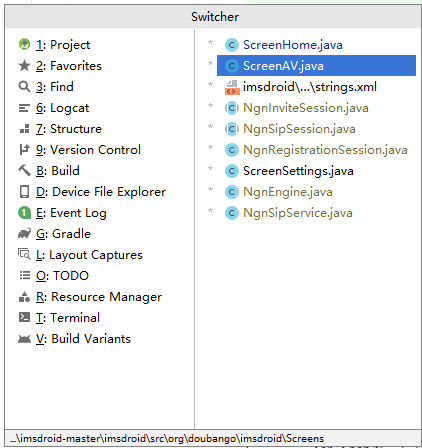


## 常规

|  |  |
| --- | --- |
| 全部保存 | Control + S |
| 同步 | Control + Alt + Y |
| 最大化/最小化编辑器 | Control + Shift + F12 |
| 添加到收藏夹 | Alt + Shift + F |
| 使用当前配置文件检查当前文件 | Alt + Shift + I |
| 快速切换模式 | Control + `（反引号） |
| 打开设置对话框 | Control + Alt + S |
| 打开项目结构对话框 | Control + Alt + Shift + S |
| 在标签和工具窗口之间切换 | Control + Tab |

## 在 Studio 内导航和搜索

在标签和工具窗口之间切换 Control + Tab



打开设置对话框 Control + Alt + S

打开项目结构对话框 Control + Alt + Shift + S

|  |  |
| --- | --- |
| 搜索全部内容（包括代码和菜单） | 按两次 Shift |
| 查找 | Control + F |
| 查找下一个 | F3 |
| 查找上一个 | Shift + F3 |
| 替换 | Control + R |
| 查找操作 | Control + Shift + A |
| 按符号名称搜索 | Control + Alt + Shift + N |
| 查找类 | Control + N |
| 查找文件（而不是类） | Control + Shift + N |
| 在路径中查找 | Control + Shift + F |
| 打开文件结构弹出式菜单 | Control + F12 |
| 在打开的编辑器标签之前导航 | Alt + 向右/向左箭头 |
| 跳到源代码 | F4 / Control + Enter |
| 在新窗口中打开当前编辑器标签 | Shift + F4 |
| 最近打开的文件弹出式菜单 | Control + E |
| 最近编辑的文件弹出式菜单 | Control + Shift + E |
| 转到上一个编辑位置 | Control + Shift + 退格键 |
| 关闭活动编辑器标签 | Control + F4 |
| 从工具窗口返回到编辑器窗口 | Esc |
| 隐藏活动或上一个活动工具窗口 | Shift + Esc |
| 转到行 | Control + G |
| 打开类型层次结构 | Control + H |
| 打开方法层次结构 | Control + Shift + H |
| 打开调用层次结构 | Control + Alt + H |

## 编写代码

|  |  |
| --- | --- |
| 生成代码（getter、setter、构造函数、hashCode/equals、toString、新文件、新类） | Alt + Insert |
| 覆盖方法 | Control + O |
| 实现方法 | Control + I |
| 控制语句 (if...else / try...catch / etc.) | Control + Alt + T |
| 删除插入符处的行 | Control + Y |
| 折叠/展开当前代码块 | Control + 减号/加号 |
| 折叠/展开所有代码块 | Control + Shift + 减号/加号 |
| 复制当前行或选择 | Control + D |
| 基本代码自动完成 | Control + 空格键 |
| 智能代码自动完成（按预期类型过滤方法和变量列表） | Control + Shift + 空格键 |
| 完成语句 | Control + Shift + Enter |
| 显示文档说明 | Control + Q |
| 显示选定方法的参数 | Control + P |
| 转到声明（直接） | Control + B 或 Control + 点击 |
| 转到实现 | Control + Alt + B |
| 转到超类方法/超类 | Control + U |
| 打开快速定义查找 | Control + Shift + I |
| 切换项目工具窗口的可见性 | Alt + 1 |
| 把光标当前位置添加书签或取消 | F11 |
| 把光标当前位置通数字标志添加或删除换书签  跳转到指定数字标志书签位置  显示当前文件的书签 | Control + F11  Ctrl + 0 ~ 9  Shift + F11 |
| 通过行注释添加注释/取消注释 | Control + / |
| 通过块注释添加注释/取消注释 | Control + Shift + / |
| 选择连续增加的代码块 | Control + W |
| 将当前选择退回到上一个状态 | Control + Shift + W |
| 移动到代码块起始位置 | Control + [ |
| 移动到代码块结束位置 | Control + ] |
| 从当前位置选择到代码块起始位置 | Control + Shift + [ |
| 从当前位置选择到代码块结束位置 | Control + Shift + ] |
| 从当前位置删除到单词结尾 | Control + Delete |
| 从当前位置删除到单词开头 | Control + 退格键 |
| 优化导入 | Control + Alt + O |
| 项目快速修复（显示意图操作和快速修复） | Alt + Enter |
| 重新格式化代码 | Control + Alt + L |
| 自动缩进行 | Control + Alt + I |
| 缩进/取消缩进行 | Tab/Shift + Tab |
| 智能行合并 | Control + Shift + J |
| 智能行拆分 | Control + Enter |
| 开始新行 | Shift + Enter |
| 下一个/上一个突出显示的错误 | F2 / Shift + F2 |

## 构建并运行

|  |  |
| --- | --- |
| 构建 | Control + F9 |
| 构建并运行 | Shift + F10 |

## 调试

|  |  |  |
| --- | --- | --- |
| 调试 | Shift + F9 | Control + D |
| 跳过 | F8 | F8 |
| 逐步执行 | F7 | F7 |
| 智能逐步执行 | Shift + F7 | Shift + F7 |
| 跳出 | Shift + F8 | Shift + F8 |
| 运行到光标位置 | Alt + F9 | Option + F9 |
| 评估表达式 | Alt + F8 | Option + F8 |
| 继续运行程序 | F9 | Command + Option + R |
| 切换断点 | Control + F8 | Command + F8 |
| 查看断点 | Control + Shift + F8 | Command + Shift + F8 |

## 重构

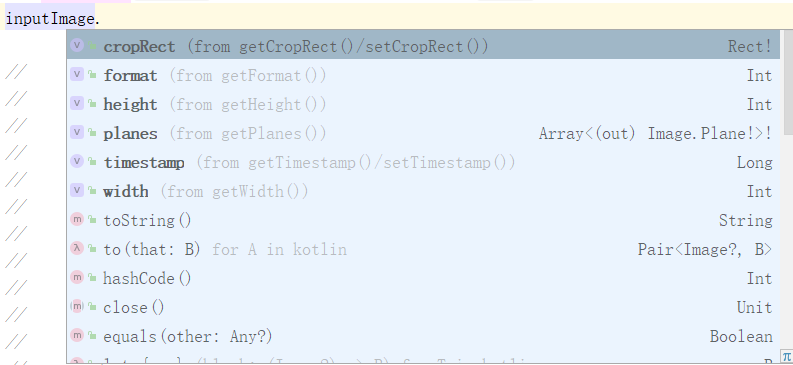
|  |  |
| --- | --- |
| 复制 | F5 |
| 移动 | F6 |
| 安全删除 | Alt + Delete |
| 重命名 | Shift + F6 |
| 更改签名 | Control + F6 |
| 内联 | Control + Alt + N |
| 提取方法 | Control + Alt + M |
| 提取变量 | Control + Alt + V |
| 提取字段 | Control + Alt + F |
| 提取常量 | Control + Alt + C |
| 提取参数 | Control + Alt + P |

## 版本控制/本地历史记录

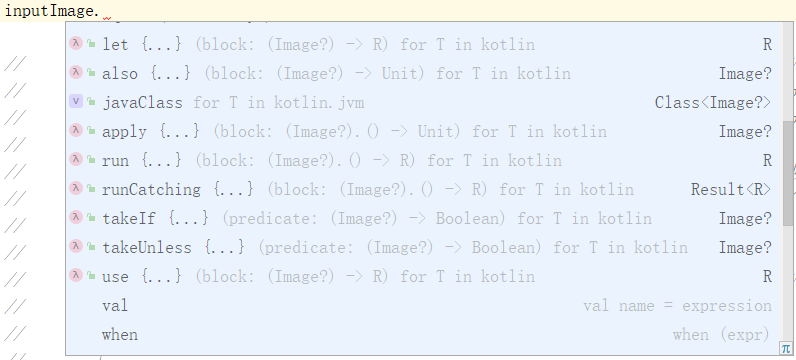
|  |  |
| --- | --- |
| 将项目提交到 VCS | Control + K |
| 从 VCS 更新项目 | Control + T |
| 查看最近变更 | Alt + Shift + C |
| 打开 VCS 弹出式菜单 | Alt + `（反引号） |

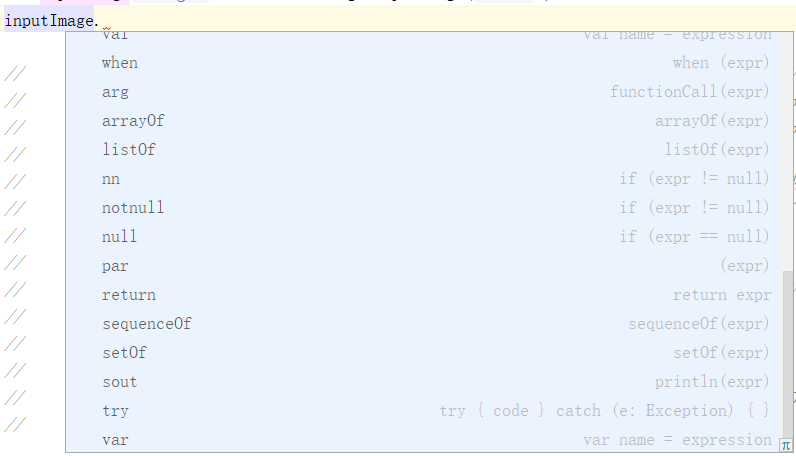
# 点方式的快捷输入

在变量或表达式后输入点，然后会有很多的选项，如下：



前面是对象的函数与属性，再往后拉就能看到快捷输入，如下：





在右侧显示了这些快捷输入的样子

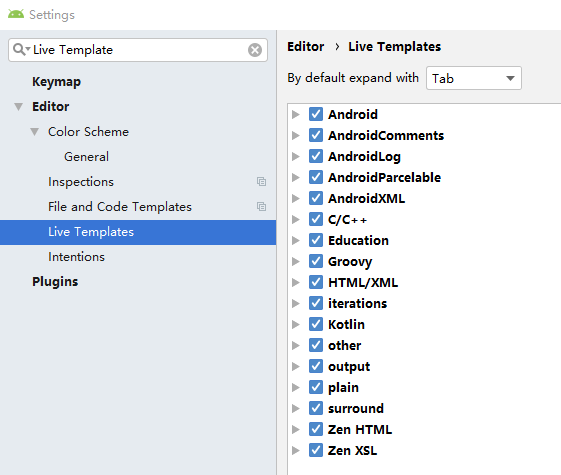
比如我们输入.sout，如下：



回车后效果如下：

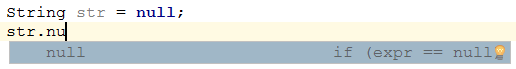


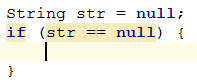
# 代码模板



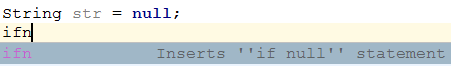
## 空判断

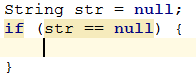
模板1：





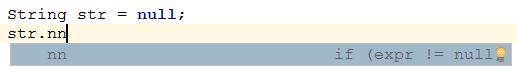
模板2：（if null）

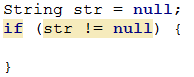




## 2、非空判断

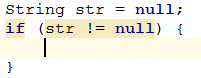
模板1：（not null）





模板2：（if not null）





## 3、抛出一个异常生成常量





## 4、查找控件

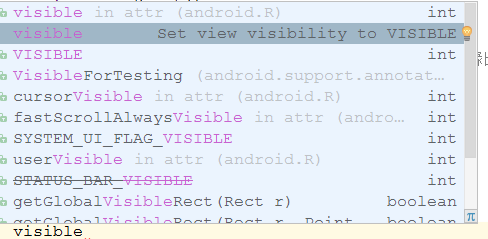




## 5、设置为GONE、visible



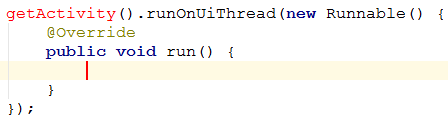




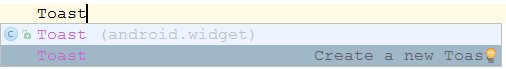


## 6、运行到UI线程





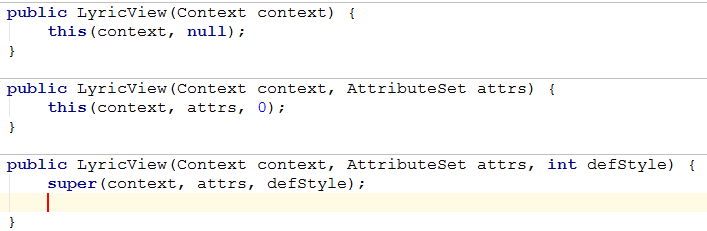
## 7、显示Toast





## 8、给View添加构造方法





## 9、TODO



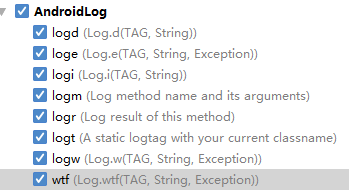


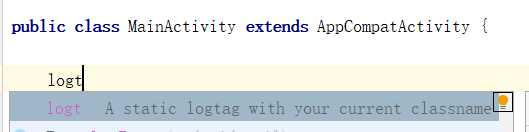
## 10、FIXME

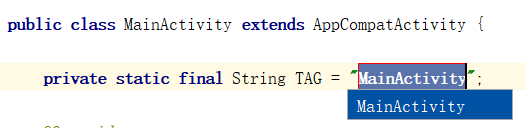




## 11、输出语句System.out\Log







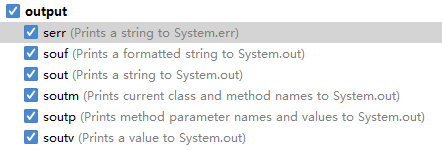




输出方法名：



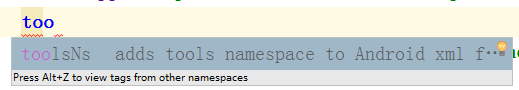




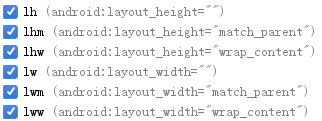
## 12、xml模板







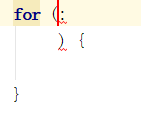




## 14、循环

创建foreach循环：





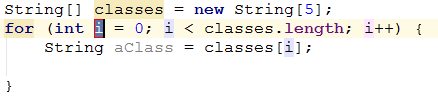
创建迭代循环





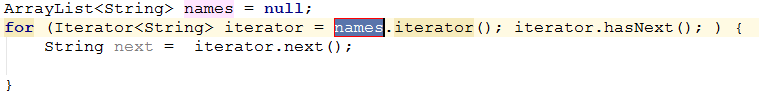
迭代数组元素（itaterate array）





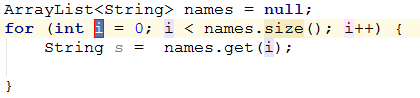
迭代集合元素（itaterate collection）





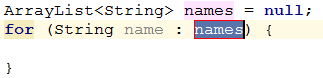
迭代List元素（itaterate list）



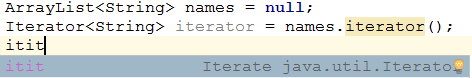


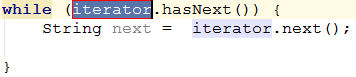
增强for循环（迭代可迭代 iterate iterable）





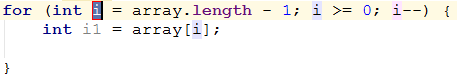
迭代一个迭代器（iterate iterable）





反过来迭代数组（reverse iterate array）





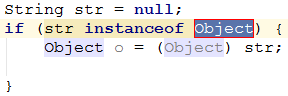
## 15、获取单例方法





## 16、类型判断（instanceof）



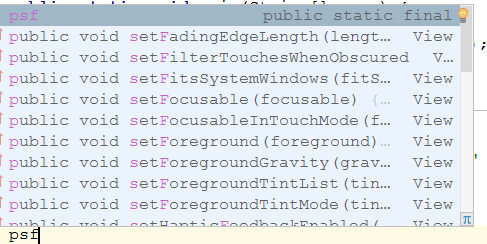


## 17、main方法（public static void main）

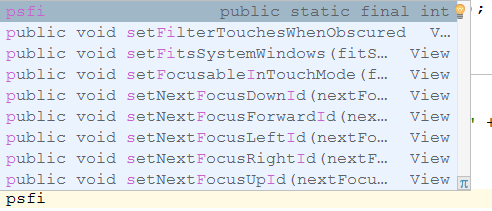




## 18、变量声明



















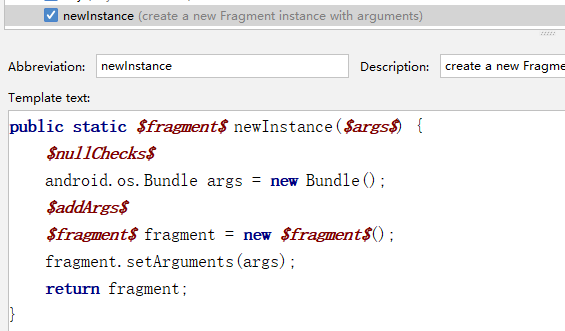


## 19、创建常量key

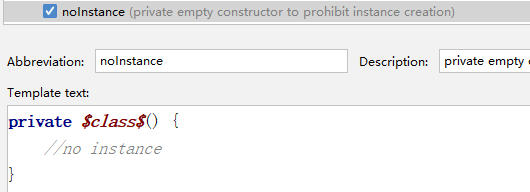




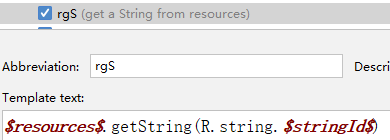
## 20、Fragment的newInstance



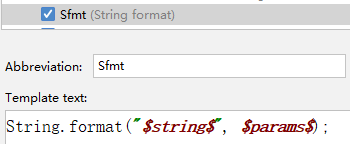
## 21、私有构造函数

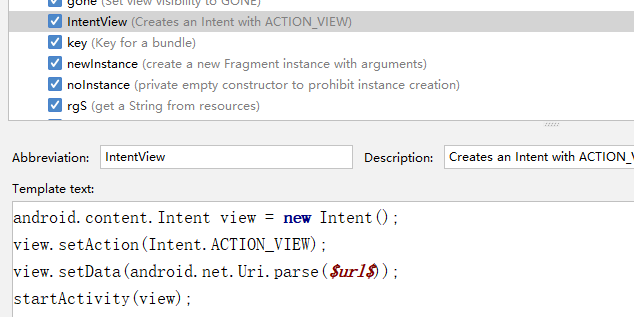


## 22、获取资源String

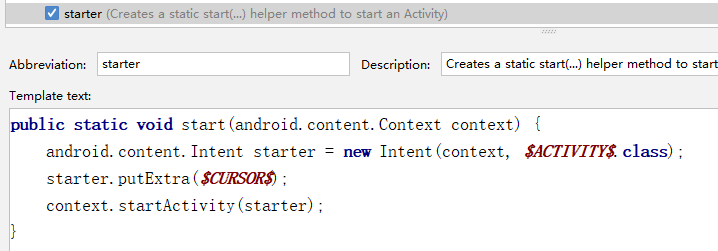


## 23、格式化String

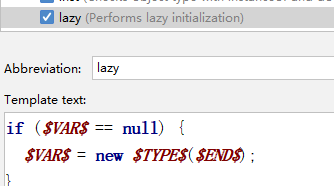




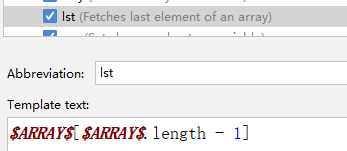
## 24、开启Activity



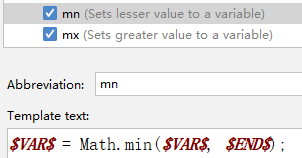
## 25、懒初始化



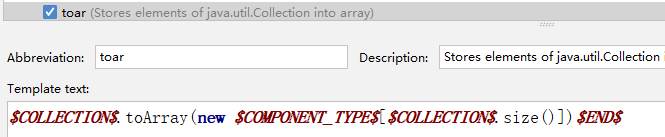
## 26、访问数组最后一个元素



## 27、最大/小值



## 28、集合转数组



# 其它技巧

1、快速生成get/set方法

