1) Filenames
   - underscore_lowercase.extension (view_page_signup.xml)
   - type_subtype_description (model_profile_user.xml)
   - Java class should follow java convention (MyClass.java)
2) Code
   - Constant name is UPPCASE_UNDERSCORE
   - Variable name is camelCase
   - Class name is CapitalCamelCase
   - Listlike name is plural, and element name is single
     - objects = [obj, obj, obj]
     - for obj in objects{ // code }
   - Don't use ++ or --, use += 1 and -= 1
   - Don't write long functions (more than 50 lines)
   - Don't use single letter variable outside a for-loop
   - Don't omit {} in control statement.
   - Don't write 3 layers of nested if
   - Don't use global variables, use singleton objects
   - Use Android Studio IDE default tab indentation
   - Recommended: Assignment is left to right, and comparison is right to left
     - is_cat_lover = true
     - if (3 == number_of_cats){ // code }
   - Lines should not be longer than 80 character.
   - Anything in a {} must be indented accordingly
   - Open bracket { is at end of a statement, that is
     - if(is_cat_lover){
     - }
   - If a if statement can be a single line, make it so
   - Align = with tab. That is:
     - variable = value
     - longVariable = value2
     - sVar = value3
   - All public data member must be private with its own getter and setter to avoid unsafe modification(Encapsulation)
3) Git commit messages
   - Verb by description
     - Adds frame to profile photo and modifies profile photo size
   - Bug fix format: Patches a bug that ____ when ____, ___ fixed it
     - Patches a bug that crush the profile layout when user click invite button, limiting the button size fixed it.

4) Documentation

- Format:

```
/**
 * This function takes ___ and does ____, then returns ____.
 *    argument1: description for why this functions needs it
 *    argument2: description for why this functions needs it
 * Return
 *    data: description of expected return
 * Throws
 *    ___exception: description of throwable
 */
return_tyep functio_nanme(argument){
     // Code
}
```

- Example

```
/**
 * This function takes 2 doubles and divides the first by the second,
 * then returns the Quotient.
 *    argument1: dividend
 *    argument2: divisor
 * Return
 *    data: quotient
 * Throws
 *    IllegalArgumentException: when divisor is zero
 */
double my_divide(double dividend, double divisor){
     if(0 == divisor){
          throw new IllegalArgumentException("'divisor' is 0");
     }
     return dividend / divisor;
}
```