得到List泛型的类型

```
public class GetGeneric {
    List<String> list = Lists.newArrayList();
    public static void main(String[] args) throws Exception{

        Class<?> clazz = GetGeneric.class;

        // 得到所有的fields
        Field[] fs = clazz.getDeclaredFields();
        for(Field f : fs) {
            // 得到field的class及类型全路径
           Class fieldClazz = f.getType();
            System.out.println(fieldClazz.getName());
            //~out:java.util.List

            //判断是否为基本类型
           if(fieldClazz.isPrimitive())
                continue;

            //getName()返回field的类型全路径
           if(fieldClazz.getName().startsWith("java.lang"))
                continue;
            if(fieldClazz.isAssignableFrom(List.class)) {
                //关键的地方，如果是List类型，得到其Generic的类型
              Type fc = f.getGenericType();
                if(fc == null) continue;
                //如果是泛型参数的类型
              if(fc instanceof ParameterizedType) {
                    ParameterizedType pt = (ParameterizedType) fc;
                    //得到泛型里的class类型对象
                  Class genericClazz =
(Class)pt.getActualTypeArguments()[0];
                    System.out.println(genericClazz.getName());
                    //~泛型类型out:java.lang.String
                }
            }
```

```
        }
    }
}
```

List<Integer>添加Object类型

```
public class TestA {
    public static void main(String[] args) throws Exception{
        List<Integer> list = new ArrayList<Integer>();
        list.add(1);
        list.add(2);
        Method method = list.getClass().getMethod("add",
Object.class);
        method.invoke(list, "List<Integer>添加Object类型");
        for (int i=0;i<list.size();i++) {
            System.out.print(list.get(i) + ", ");
            //~out: 1, 2, List<Integer>添加Object类型,
        }
    }
}
```