金盾项目，手写笔没有外圈圆变化：

数据如下：

EV\_ABS ABS\_PRESSURE 000003aa

EV\_SYN SYN\_REPORT 00000000

EV\_ABS ABS\_X 00000236

EV\_ABS ABS\_Y 0000036d

EV\_ABS ABS\_PRESSURE 000003ac

EV\_SYN SYN\_REPORT 00000000

EV\_ABS ABS\_X 00000237

EV\_ABS ABS\_Y 00000376

EV\_ABS ABS\_PRESSURE 000003ae

EV\_SYN SYN\_REPORT 00000000

EV\_ABS ABS\_Y 0000037f

EV\_ABS ABS\_PRESSURE 000003b1

EV\_SYN SYN\_REPORT 00000000

EV\_ABS ABS\_X 00000238

EV\_ABS ABS\_Y 00000387

EV\_ABS ABS\_PRESSURE 000003b4

EV\_SYN SYN\_REPORT 00000000

EV\_ABS ABS\_X 00000239

EV\_ABS ABS\_Y 0000038d

EV\_ABS ABS\_PRESSURE 000003b7

EV\_SYN SYN\_REPORT 00000000

EV\_ABS ABS\_Y 00000393

EV\_ABS ABS\_PRESSURE 000003bf

EV\_SYN SYN\_REPORT 00000000

EV\_ABS ABS\_X 0000023a

EV\_ABS ABS\_Y 0000039a

EV\_ABS ABS\_PRESSURE 000003c2

EV\_SYN SYN\_REPORT 00000000

EV\_ABS ABS\_X 0000023b

EV\_ABS ABS\_Y 000003a3

EV\_ABS ABS\_PRESSURE 000003c7

EV\_SYN SYN\_REPORT 00000000

EV\_ABS ABS\_Y 000003ad

EV\_ABS ABS\_PRESSURE 000003cb

EV\_SYN SYN\_REPORT 00000000

EV\_ABS ABS\_Y 000003b5

EV\_ABS ABS\_PRESSURE 000003cf

EV\_SYN SYN\_REPORT 00000000

EV\_ABS ABS\_X 0000023c

EV\_ABS ABS\_Y 000003bd

EV\_ABS ABS\_PRESSURE 000003d1

EV\_SYN SYN\_REPORT 00000000

EV\_ABS ABS\_Y 000003c4

EV\_ABS ABS\_PRESSURE 000003d4

EV\_SYN SYN\_REPORT 00000000

EV\_ABS ABS\_X 0000023d

EV\_ABS ABS\_Y 000003cc

EV\_ABS ABS\_PRESSURE 000003d6

EV\_SYN SYN\_REPORT 00000000

EV\_ABS ABS\_Y 000003d6

EV\_ABS ABS\_PRESSURE 000003d9

EV\_SYN SYN\_REPORT 00000000

EV\_ABS ABS\_X 0000023e

EV\_ABS ABS\_Y 000003df

EV\_ABS ABS\_PRESSURE 000003db

EV\_SYN SYN\_REPORT 00000000

EV\_ABS ABS\_Y 000003e8

EV\_ABS ABS\_PRESSURE 000003dd

EV\_SYN SYN\_REPORT 00000000

EV\_ABS ABS\_X 0000023f

EV\_ABS ABS\_Y 000003ef

EV\_ABS ABS\_PRESSURE 000003de

EV\_SYN SYN\_REPORT 00000000

EV\_ABS ABS\_Y 000003f7

EV\_SYN SYN\_REPORT 00000000

EV\_ABS ABS\_X 00000240

EV\_ABS ABS\_Y 00000400

EV\_ABS ABS\_PRESSURE 000003df

EV\_SYN SYN\_REPORT 00000000

EV\_ABS ABS\_Y 00000409

EV\_ABS ABS\_PRESSURE 000003e1

EV\_SYN SYN\_REPORT 00000000

EV\_ABS ABS\_X 00000241

EV\_ABS ABS\_Y 00000411

EV\_ABS ABS\_PRESSURE 000003e3

EV\_SYN SYN\_REPORT 00000000

EV\_ABS ABS\_Y 00000419

EV\_ABS ABS\_PRESSURE 000003e8

EV\_SYN SYN\_REPORT 00000000

EV\_ABS ABS\_Y 0000041f

EV\_ABS ABS\_PRESSURE 000003eb

EV\_SYN SYN\_REPORT 00000000

EV\_ABS ABS\_X 00000242

EV\_ABS ABS\_Y 00000427

乐凡的可以显示变化的外圆:

数据如下:

EV\_ABS ABS\_MT\_SLOT 00000009

EV\_ABS ABS\_MT\_TRACKING\_ID 00000009

EV\_ABS ABS\_MT\_POSITION\_X 00000335

EV\_ABS ABS\_MT\_POSITION\_Y 0000034a

EV\_SYN SYN\_REPORT 00000000

EV\_ABS ABS\_MT\_PRESSURE 00000090

EV\_ABS ABS\_MT\_POSITION\_X 00000336

EV\_ABS ABS\_MT\_TOUCH\_MAJOR 00000090

EV\_ABS ABS\_MT\_WIDTH\_MAJOR 00000090

EV\_SYN SYN\_REPORT 00000000

EV\_ABS ABS\_MT\_PRESSURE 000000a4

EV\_ABS ABS\_MT\_POSITION\_X 00000338

EV\_ABS ABS\_MT\_POSITION\_Y 00000349

EV\_ABS ABS\_MT\_TOUCH\_MAJOR 000000a4

EV\_ABS ABS\_MT\_WIDTH\_MAJOR 000000a4

EV\_SYN SYN\_REPORT 00000000

EV\_ABS ABS\_MT\_PRESSURE 000000b6

EV\_ABS ABS\_MT\_POSITION\_X 0000033b

EV\_ABS ABS\_MT\_POSITION\_Y 00000347

EV\_ABS ABS\_MT\_TOUCH\_MAJOR 000000b6

EV\_ABS ABS\_MT\_WIDTH\_MAJOR 000000b6

EV\_SYN SYN\_REPORT 00000000

EV\_ABS ABS\_MT\_PRESSURE 000000c7

EV\_ABS ABS\_MT\_POSITION\_X 0000033d

EV\_ABS ABS\_MT\_POSITION\_Y 00000346

EV\_ABS ABS\_MT\_TOUCH\_MAJOR 000000c7

EV\_ABS ABS\_MT\_WIDTH\_MAJOR 000000c7

EV\_SYN SYN\_REPORT 00000000

EV\_ABS ABS\_MT\_PRESSURE 000000d8

EV\_ABS ABS\_MT\_POSITION\_X 0000033f

EV\_ABS ABS\_MT\_POSITION\_Y 00000344

EV\_ABS ABS\_MT\_TOUCH\_MAJOR 000000d8

EV\_ABS ABS\_MT\_WIDTH\_MAJOR 000000d8

EV\_SYN SYN\_REPORT 00000000

EV\_ABS ABS\_MT\_PRESSURE 000000ea

EV\_ABS ABS\_MT\_POSITION\_X 00000340

EV\_ABS ABS\_MT\_POSITION\_Y 00000342

EV\_ABS ABS\_MT\_TOUCH\_MAJOR 000000ea

EV\_ABS ABS\_MT\_WIDTH\_MAJOR 000000ea

EV\_SYN SYN\_REPORT 00000000

EV\_ABS ABS\_MT\_PRESSURE 000000fd

EV\_ABS ABS\_MT\_POSITION\_X 00000341

EV\_ABS ABS\_MT\_POSITION\_Y 00000340

EV\_ABS ABS\_MT\_TOUCH\_MAJOR 000000fd

EV\_ABS ABS\_MT\_WIDTH\_MAJOR 000000fd

EV\_SYN SYN\_REPORT 00000000

EV\_ABS ABS\_MT\_PRESSURE 00000111

EV\_ABS ABS\_MT\_POSITION\_X 00000342

EV\_ABS ABS\_MT\_POSITION\_Y 0000033d

EV\_ABS ABS\_MT\_TOUCH\_MAJOR 00000111

EV\_ABS ABS\_MT\_WIDTH\_MAJOR 00000111

EV\_SYN SYN\_REPORT 00000000

EV\_ABS ABS\_MT\_PRESSURE 00000127

EV\_ABS ABS\_MT\_POSITION\_X 00000343

EV\_ABS ABS\_MT\_POSITION\_Y 0000033b

EV\_ABS ABS\_MT\_TOUCH\_MAJOR 00000127

EV\_ABS ABS\_MT\_WIDTH\_MAJOR 00000127

EV\_SYN SYN\_REPORT 00000000

EV\_ABS ABS\_MT\_PRESSURE 0000013c

EV\_ABS ABS\_MT\_POSITION\_Y 00000339

EV\_ABS ABS\_MT\_TOUCH\_MAJOR 0000013c

EV\_ABS ABS\_MT\_WIDTH\_MAJOR 0000013c

EV\_SYN SYN\_REPORT 00000000

EV\_ABS ABS\_MT\_PRESSURE 00000152

EV\_ABS ABS\_MT\_POSITION\_Y 00000337

EV\_ABS ABS\_MT\_TOUCH\_MAJOR 00000152

EV\_ABS ABS\_MT\_WIDTH\_MAJOR 00000152

EV\_SYN SYN\_REPORT 00000000

EV\_ABS ABS\_MT\_PRESSURE 00000167

EV\_ABS ABS\_MT\_POSITION\_X 00000344

EV\_ABS ABS\_MT\_POSITION\_Y 00000335

EV\_ABS ABS\_MT\_TOUCH\_MAJOR 00000167

EV\_ABS ABS\_MT\_WIDTH\_MAJOR 00000167

EV\_SYN SYN\_REPORT 00000000

EV\_ABS ABS\_MT\_PRESSURE 0000017c

EV\_ABS ABS\_MT\_POSITION\_Y 00000332

EV\_ABS ABS\_MT\_TOUCH\_MAJOR 0000017c

EV\_ABS ABS\_MT\_WIDTH\_MAJOR 0000017c

EV\_SYN SYN\_REPORT 00000000

EV\_ABS ABS\_MT\_PRESSURE 0000018e

EV\_ABS ABS\_MT\_POSITION\_X 00000345

EV\_ABS ABS\_MT\_POSITION\_Y 00000330

EV\_ABS ABS\_MT\_TOUCH\_MAJOR 0000018e

EV\_ABS ABS\_MT\_WIDTH\_MAJOR 0000018e

EV\_SYN SYN\_REPORT 00000000

EV\_ABS ABS\_MT\_PRESSURE 0000019f

EV\_ABS ABS\_MT\_POSITION\_X 00000346

EV\_ABS ABS\_MT\_POSITION\_Y 0000032e

EV\_ABS ABS\_MT\_TOUCH\_MAJOR 0000019f

EV\_ABS ABS\_MT\_WIDTH\_MAJOR 0000019f

EV\_SYN SYN\_REPORT 00000000

EV\_ABS ABS\_MT\_PRESSURE 000001ae

EV\_ABS ABS\_MT\_POSITION\_X 00000347

EV\_ABS ABS\_MT\_POSITION\_Y 0000032c

EV\_ABS ABS\_MT\_TOUCH\_MAJOR 000001ae

EV\_ABS ABS\_MT\_WIDTH\_MAJOR 000001ae

EV\_SYN SYN\_REPORT 00000000

EV\_ABS ABS\_MT\_PRESSURE 000001b9

EV\_ABS ABS\_MT\_POSITION\_X 00000348

EV\_ABS ABS\_MT\_POSITION\_Y 0000032a

EV\_ABS ABS\_MT\_TOUCH\_MAJOR 000001b9

EV\_ABS ABS\_MT\_WIDTH\_MAJOR 000001b9

EV\_SYN SYN\_REPORT 00000000

EV\_ABS ABS\_MT\_PRESSURE 000001c5

EV\_ABS ABS\_MT\_POSITION\_X 00000349

EV\_ABS ABS\_MT\_POSITION\_Y 00000328

EV\_ABS ABS\_MT\_TOUCH\_MAJOR 000001c5

EV\_ABS ABS\_MT\_WIDTH\_MAJOR 000001c5

EV\_SYN SYN\_REPORT 00000000

EV\_ABS ABS\_MT\_PRESSURE 000001d1

EV\_ABS ABS\_MT\_POSITION\_X 0000034b

EV\_ABS ABS\_MT\_POSITION\_Y 00000325

EV\_ABS ABS\_MT\_TOUCH\_MAJOR 000001d1

EV\_ABS ABS\_MT\_WIDTH\_MAJOR 000001d1

EV\_SYN SYN\_REPORT 00000000

EV\_ABS ABS\_MT\_PRESSURE 000001df

EV\_ABS ABS\_MT\_POSITION\_X 0000034c

EV\_ABS ABS\_MT\_POSITION\_Y 00000322

EV\_ABS ABS\_MT\_TOUCH\_MAJOR 000001df

EV\_ABS ABS\_MT\_WIDTH\_MAJOR 000001df

EV\_SYN SYN\_REPORT 00000000

EV\_ABS ABS\_MT\_PRESSURE 000001f0

EV\_ABS ABS\_MT\_POSITION\_X 0000034d

EV\_ABS ABS\_MT\_POSITION\_Y 0000031e

EV\_ABS ABS\_MT\_TOUCH\_MAJOR 000001f0

EV\_ABS ABS\_MT\_WIDTH\_MAJOR 000001f0

EV\_SYN SYN\_REPORT 00000000

EV\_ABS ABS\_MT\_PRESSURE 00000202

EV\_ABS ABS\_MT\_POSITION\_X 0000034e

EV\_ABS ABS\_MT\_POSITION\_Y 0000031a

EV\_ABS ABS\_MT\_TOUCH\_MAJOR 00000202

EV\_ABS ABS\_MT\_WIDTH\_MAJOR 00000202

EV\_SYN SYN\_REPORT 00000000

EV\_ABS ABS\_MT\_PRESSURE 00000217

EV\_ABS ABS\_MT\_POSITION\_X 0000034f

EV\_ABS ABS\_MT\_POSITION\_Y 00000317

EV\_ABS ABS\_MT\_TOUCH\_MAJOR 00000217

EV\_ABS ABS\_MT\_WIDTH\_MAJOR 00000217

EV\_SYN SYN\_REPORT 00000000

EV\_ABS ABS\_MT\_PRESSURE 00000230

EV\_ABS ABS\_MT\_POSITION\_Y 00000314

EV\_ABS ABS\_MT\_TOUCH\_MAJOR 00000230

EV\_ABS ABS\_MT\_WIDTH\_MAJOR 00000230

EV\_SYN SYN\_REPORT 00000000

EV\_ABS ABS\_MT\_PRESSURE 0000024c

EV\_ABS ABS\_MT\_POSITION\_Y 00000311

EV\_ABS ABS\_MT\_TOUCH\_MAJOR 0000024c

EV\_ABS ABS\_MT\_WIDTH\_MAJOR 0000024c

EV\_SYN SYN\_REPORT 00000000

EV\_ABS ABS\_MT\_PRESSURE 00000268

EV\_ABS ABS\_MT\_POSITION\_Y 0000030e

EV\_ABS ABS\_MT\_TOUCH\_MAJOR 00000268

EV\_ABS ABS\_MT\_WIDTH\_MAJOR 00000268

EV\_SYN SYN\_REPORT 00000000

EV\_ABS ABS\_MT\_PRESSURE 00000280

EV\_ABS ABS\_MT\_POSITION\_Y 0000030c

EV\_ABS ABS\_MT\_TOUCH\_MAJOR 00000280

EV\_ABS ABS\_MT\_WIDTH\_MAJOR 00000280

EV\_SYN SYN\_REPORT 00000000

EV\_ABS ABS\_MT\_PRESSURE 00000294

EV\_ABS ABS\_MT\_POSITION\_X 00000350

EV\_ABS ABS\_MT\_POSITION\_Y 0000030a

EV\_ABS ABS\_MT\_TOUCH\_MAJOR 00000294

EV\_ABS ABS\_MT\_WIDTH\_MAJOR 00000294

EV\_SYN SYN\_REPORT 00000000

EV\_ABS ABS\_MT\_PRESSURE 000002a6

EV\_ABS ABS\_MT\_POSITION\_X 00000351

EV\_ABS ABS\_MT\_POSITION\_Y 00000307

EV\_ABS ABS\_MT\_TOUCH\_MAJOR 000002a6

EV\_ABS ABS\_MT\_WIDTH\_MAJOR 000002a6

EV\_SYN SYN\_REPORT 00000000

EV\_ABS ABS\_MT\_PRESSURE 000002b5

EV\_ABS ABS\_MT\_POSITION\_X 00000352

EV\_ABS ABS\_MT\_POSITION\_Y 00000304

EV\_ABS ABS\_MT\_TOUCH\_MAJOR 000002b5

EV\_ABS ABS\_MT\_WIDTH\_MAJOR 000002b5

EV\_SYN SYN\_REPORT 00000000

EV\_ABS ABS\_MT\_PRESSURE 000002c1

EV\_ABS ABS\_MT\_POSITION\_X 00000354

EV\_ABS ABS\_MT\_POSITION\_Y 00000301

EV\_ABS ABS\_MT\_TOUCH\_MAJOR 000002c1

EV\_ABS ABS\_MT\_WIDTH\_MAJOR 000002c1

EV\_SYN SYN\_REPORT 00000000

EV\_ABS ABS\_MT\_PRESSURE 000002cb

EV\_ABS ABS\_MT\_POSITION\_X 00000355

EV\_ABS ABS\_MT\_POSITION\_Y 000002fe

EV\_ABS ABS\_MT\_TOUCH\_MAJOR 000002cb

EV\_ABS ABS\_MT\_WIDTH\_MAJOR 000002cb

EV\_SYN SYN\_REPORT 00000000

EV\_ABS ABS\_MT\_PRESSURE 000002d6

EV\_ABS ABS\_MT\_POSITION\_X 00000356

EV\_ABS ABS\_MT\_POSITION\_Y 000002fb

EV\_ABS ABS\_MT\_TOUCH\_MAJOR 000002d6

EV\_ABS ABS\_MT\_WIDTH\_MAJOR 000002d6

EV\_SYN SYN\_REPORT 00000000

EV\_ABS ABS\_MT\_PRESSURE 000002e3

EV\_ABS ABS\_MT\_POSITION\_X 00000357

EV\_ABS ABS\_MT\_POSITION\_Y 000002f8

EV\_ABS ABS\_MT\_TOUCH\_MAJOR 000002e3

EV\_ABS ABS\_MT\_WIDTH\_MAJOR 000002e3

EV\_SYN SYN\_REPORT 00000000

EV\_ABS ABS\_MT\_PRESSURE 000002f4

EV\_ABS ABS\_MT\_POSITION\_X 00000358

EV\_ABS ABS\_MT\_POSITION\_Y 000002f5

EV\_ABS ABS\_MT\_TOUCH\_MAJOR 000002f4

EV\_ABS ABS\_MT\_WIDTH\_MAJOR 000002f4

EV\_SYN SYN\_REPORT 00000000

EV\_ABS ABS\_MT\_PRESSURE 00000308

EV\_ABS ABS\_MT\_POSITION\_X 00000359

EV\_ABS ABS\_MT\_POSITION\_Y 000002f3

EV\_ABS ABS\_MT\_TOUCH\_MAJOR 00000308

EV\_ABS ABS\_MT\_WIDTH\_MAJOR 00000308

EV\_SYN SYN\_REPORT 00000000

EV\_ABS ABS\_MT\_PRESSURE 0000031e

EV\_ABS ABS\_MT\_POSITION\_X 0000035a

EV\_ABS ABS\_MT\_POSITION\_Y 000002f1

EV\_ABS ABS\_MT\_TOUCH\_MAJOR 0000031e

EV\_ABS ABS\_MT\_WIDTH\_MAJOR 0000031e

EV\_SYN SYN\_REPORT 00000000

EV\_ABS ABS\_MT\_PRESSURE 00000335

EV\_ABS ABS\_MT\_POSITION\_X 0000035b

EV\_ABS ABS\_MT\_POSITION\_Y 000002ef

EV\_ABS ABS\_MT\_TOUCH\_MAJOR 00000335

EV\_ABS ABS\_MT\_WIDTH\_MAJOR 00000335

EV\_SYN SYN\_REPORT 00000000

EV\_ABS ABS\_MT\_PRESSURE 0000034b

EV\_ABS ABS\_MT\_POSITION\_Y 000002ed

EV\_ABS ABS\_MT\_TOUCH\_MAJOR 0000034b

EV\_ABS ABS\_MT\_WIDTH\_MAJOR 0000034b

EV\_SYN SYN\_REPORT 00000000

EV\_ABS ABS\_MT\_PRESSURE 0000035c

EV\_ABS ABS\_MT\_POSITION\_X 0000035c

EV\_ABS ABS\_MT\_POSITION\_Y 000002eb

EV\_ABS ABS\_MT\_TOUCH\_MAJOR 0000035c

EV\_ABS ABS\_MT\_WIDTH\_MAJOR 0000035c

EV\_SYN SYN\_REPORT 00000000

EV\_ABS ABS\_MT\_PRESSURE 00000366

EV\_ABS ABS\_MT\_POSITION\_X 0000035d

EV\_ABS ABS\_MT\_POSITION\_Y 000002e8

EV\_ABS ABS\_MT\_TOUCH\_MAJOR 00000366

EV\_ABS ABS\_MT\_WIDTH\_MAJOR 00000366

EV\_SYN SYN\_REPORT 00000000

EV\_ABS ABS\_MT\_PRESSURE 0000036d

EV\_ABS ABS\_MT\_POSITION\_X 0000035e

EV\_ABS ABS\_MT\_POSITION\_Y 000002e5

EV\_ABS ABS\_MT\_TOUCH\_MAJOR 0000036d

EV\_ABS ABS\_MT\_WIDTH\_MAJOR 0000036d

EV\_SYN SYN\_REPORT 00000000

EV\_ABS ABS\_MT\_PRESSURE 00000373

EV\_ABS ABS\_MT\_POSITION\_X 0000035f

EV\_ABS ABS\_MT\_POSITION\_Y 000002e2

EV\_ABS ABS\_MT\_TOUCH\_MAJOR 00000373

EV\_ABS ABS\_MT\_WIDTH\_MAJOR 00000373

EV\_SYN SYN\_REPORT 00000000

EV\_ABS ABS\_MT\_PRESSURE 00000376

EV\_ABS ABS\_MT\_POSITION\_X 00000360

EV\_ABS ABS\_MT\_POSITION\_Y 000002df

EV\_ABS ABS\_MT\_TOUCH\_MAJOR 00000376

EV\_ABS ABS\_MT\_WIDTH\_MAJOR 00000376

EV\_SYN SYN\_REPORT 00000000

EV\_ABS ABS\_MT\_PRESSURE 00000377

EV\_ABS ABS\_MT\_POSITION\_X 00000361

EV\_ABS ABS\_MT\_POSITION\_Y 000002dc

EV\_ABS ABS\_MT\_TOUCH\_MAJOR 00000377

EV\_ABS ABS\_MT\_WIDTH\_MAJOR 00000377

EV\_SYN SYN\_REPORT 00000000

EV\_ABS ABS\_MT\_PRESSURE 00000378

EV\_ABS ABS\_MT\_POSITION\_X 00000362

EV\_ABS ABS\_MT\_POSITION\_Y 000002da

EV\_ABS ABS\_MT\_TOUCH\_MAJOR 00000378

EV\_ABS ABS\_MT\_WIDTH\_MAJOR 00000378

EV\_SYN SYN\_REPORT 00000000

EV\_ABS ABS\_MT\_POSITION\_X 00000363

EV\_ABS ABS\_MT\_POSITION\_Y 000002d7

EV\_SYN SYN\_REPORT 00000000

EV\_ABS ABS\_MT\_PRESSURE 0000037a

EV\_ABS ABS\_MT\_POSITION\_X 00000364

EV\_ABS ABS\_MT\_POSITION\_Y 000002d6

EV\_ABS ABS\_MT\_TOUCH\_MAJOR 0000037a

EV\_ABS ABS\_MT\_WIDTH\_MAJOR 0000037a

EV\_SYN SYN\_REPORT 00000000

EV\_ABS ABS\_MT\_PRESSURE 0000037d

EV\_ABS ABS\_MT\_POSITION\_X 00000365

EV\_ABS ABS\_MT\_POSITION\_Y 000002d4

EV\_ABS ABS\_MT\_TOUCH\_MAJOR 0000037d

EV\_ABS ABS\_MT\_WIDTH\_MAJOR 0000037d

EV\_SYN SYN\_REPORT 00000000

EV\_ABS ABS\_MT\_PRESSURE 00000383

EV\_ABS ABS\_MT\_POSITION\_Y 000002d2

EV\_ABS ABS\_MT\_TOUCH\_MAJOR 00000383

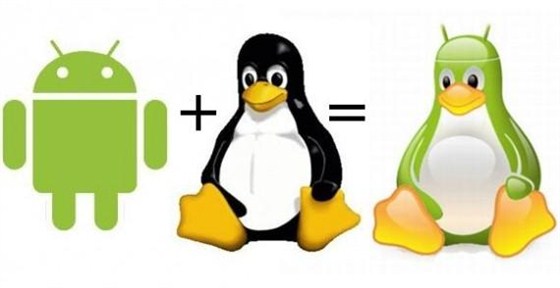
EV\_ABS ABS\_MT\_WIDTH\_MAJOR 00000383

# 触摸屏技术问题-Linux与Android 多点触摸协议（非技术看不懂）

2017-12-27 19:35:00

大家都知道Android是基于Linux内核的操作系统，也曾经和Linux基金会因为内核问题产生过分歧，本文将开始对Android的内核进行剖析，主要介绍Android和Linux之间的关系，后续还会讲到Android系统在Linux系统之上扩展的部分功能和驱动。  
虽然Android基于Linux内核，但是它与Linux之间还是有很大的差别，比如Android在Linux内核的基础上添加了自己所特有的驱动程序。下面我们就来分析一下它们之间究竟有什么关系?

android是否能称为一种新的操作系统呢？至少我自己认为不算是，它多算作一个新的应用程序罢了。



为了使用功能强大的多点触控设备，就需要一种方案去上报用户层所需的详细的手指触摸数据。这个文档所描述的多点触控协议可以让内核驱动程序向用户层上报任意多指的数据信息。

使用说明

单点触摸信息是以ABS承载并按一定顺序发送，如BTN\_TOUCH、ABS\_X、ABS\_Y、SYNC。而多点触摸信息则是以ABS\_MT承载并按一定顺序发送，如ABS\_MT\_POSITION\_X、ABS\_MT\_POSITION\_Y，然后通过调用input\_mt\_sync（）产生一个 SYN\_MT\_REPORT event来标记一个点的结束，告诉接收方接收当前手指的信息并准备接收其它手指的触控信息。后调用 input\_sync（）函数上报触摸信息开始动作并告诉接收方开始接收下一系列多点触摸信息。

协议定义了一系列ABS\_MT事件，这些事件被分为几大类，充许只应用其中的一部份，多点触摸小的事件集中应包括ABS\_MT\_TOUCH\_MAJOR、ABS\_MT\_POSITION\_X和 ABS\_MT\_POSITION\_Y，以此来实现多点触摸。如果设备支持ABS\_MT\_WIDTH\_MAJOR这个事件，那么此事件可以提供手指触摸接触面积大小。触摸方向等信息可以由ABS\_MT\_TOUCH\_MINOR, ABS\_MT\_WIDTH\_MINOR and ABS\_MT\_ORIENTATION提供。ABS\_MT\_TOOL\_TYPE提供触摸设备的类别，如手或是笔或是其它。后有些设备可能会支持ABS\_MT\_TRACKING\_ID，用来支持硬件跟踪多点信息，即该点属于哪一条线等。

下面是两点触摸支持的小事件集序列：

ABS\_MT\_TOUCH\_MAJOR

ABS\_MT\_POSITION\_X

ABS\_MT\_POSITION\_Y

SYN\_MT\_REPORT //上报个点

ABS\_MT\_TOUCH\_MAJOR

ABS\_MT\_POSITION\_X

ABS\_MT\_POSITION\_Y

SYN\_MT\_REPORT //上报个点

SYN\_REPORT //开始动作

Event 原语

“接触”一词用来描述一个物体直接碰到另一个物体的表面。

ABS\_MT\_TOUCH\_MAJOR描述了主接触面的长轴，它和X，Y同一个单位，如果一个面的分辨率为X\*Y，则ABS\_MT\_TOUCH\_MAJOR的大值为sqrt（X^2+Y^2）

ABS\_MT\_TOUCH\_MINOR描述了接触面的短轴，如果接触面是圆形，它可以不用。

ABS\_MT\_WIDTH\_MAJOR描述了接触工具的长轴

ABS\_MT\_WIDTH\_MINOR描述了接触工具的短轴

ABS\_MT\_TOUCH\_MAJOR := max（X, Y）

ABS\_MT\_TOUCH\_MINOR := min（X, Y）

ABS\_MT\_ORIENTATION := bool（X > Y）

以上四个参数可以用来生成额外的触摸信息，如ABS\_MT\_TOUCH\_MAJOR/ABS\_MT\_WIDTH\_MAJOR的比率可以用来描述压力。

ABS\_MT\_ORIENTATION

ABS\_MT\_POSITION\_X接触面的点X坐标

ABS\_MT\_POSITION\_Y接触面的点Y坐标

ABS\_MT\_TOOL\_TYPE描述接触工具类型，很多内核驱动无法区分此参数如手指及笔，如果是这样，该参数可以不用，协议目前支持MT\_TOOL\_FINGER和MT\_TOOL\_PEN两种类型。

ABS\_MT\_BLOB\_ID形状集ID，集合几个点以描述一个形状，很多驱动没有形状属性，此参数可以不用。

ABS\_MT\_TRACKING\_ID描述了从接触开始到释放的整个过程的集合，如果设备不支持，此参数可是不用。

触摸轨迹

仅有少数设备可以明触的标识真实的 trackingID，多数情况下 trackingID只能来标识一次触摸动作的过程。

手势

多点触摸指定的应用是创建手势动作， TOUCH和 WIDTH参数经常用来区别手指的压力和手指间的距离，另外 MINOR类的参数可以用来区别设备的接触面的大小（点接触还是面接触），ORIENTATION可以产生旋转事件。

在Linux内核支持的基础上，Android在其2.0源码中加入多点触摸功能。由此触摸屏在Android的frameworks被完全分为2种实现途径：单点触摸屏的单点方式，多点触摸屏的单点和多点方式。

在Linux的input.h中，多点触摸功能依赖于以下几个主要的软件位：

……

#define SYN\_REPORT0

#define SYN\_CONFIG1

#define SYN\_MT\_REPORT2

……

#define ABS\_MT\_TOUCH\_MAJOR0x30

#define ABS\_MT\_TOUCH\_MINOR0x31

#define ABS\_MT\_WIDTH\_MAJOR0x32

#define ABS\_MT\_WIDTH\_MINOR0x33

#define ABS\_MT\_ORIENTATION0x34

#define ABS\_MT\_POSITION\_X0x35

#define ABS\_MT\_POSITION\_Y0x36

#define ABS\_MT\_TOOL\_TYPE0x37

#define ABS\_MT\_BLOB\_ID0x38

……

在Android中对应的软件位定义在RawInputEvent.java中：

……

public class RawInputEvent {

……

public static final int CLASS\_TOUCHSCREEN\_MT = 0x00000010;

……

public static final int ABS\_MT\_TOUCH\_MAJOR = 0x30;

public static final int ABS\_MT\_TOUCH\_MINOR = 0x31;

public static final int ABS\_MT\_WIDTH\_MAJOR = 0x32;

public static final int ABS\_MT\_WIDTH\_MINOR = 0x33;

public static final int ABS\_MT\_ORIENTATION = 0x34;

public static final int ABS\_MT\_POSITION\_X = 0x35;

public static final int ABS\_MT\_POSITION\_Y = 0x36;

public static final int ABS\_MT\_TOOL\_TYPE = 0x37;

public static final int ABS\_MT\_BLOB\_ID = 0x38;

……

public static final int SYN\_REPORT = 0;

public static final int SYN\_CONFIG = 1;

public static final int SYN\_MT\_REPORT = 2;

……

在Android中，多点触摸的实现方法在具体的代码实现中和单点是完全区分开的。在Android代码的EventHub.cpp中，单点屏和多点屏由如下代码段来判定：

int EventHub::open\_device（const char \*deviceName）

{

……

if （test\_bit（ABS\_MT\_TOUCH\_MAJOR, abs\_bitmask）

&& test\_bit（ABS\_MT\_POSITION\_X, abs\_bitmask）

&& test\_bit（ABS\_MT\_POSITION\_Y, abs\_bitmask）） {

device->classes |= CLASS\_TOUCHSCREEN | CLASS\_TOUCHSCREEN\_MT;

//LOGI（"It is a multi-touch screen!"）；

}

//single-touch?

else if （test\_bit（BTN\_TOUCH, key\_bitmask）

&& test\_bit（ABS\_X, abs\_bitmask）

&& test\_bit（ABS\_Y, abs\_bitmask）） {

device->classes |= CLASS\_TOUCHSCREEN;

//LOGI（"It is a single-touch screen!"）；

}

……

}

我们知道，在触摸屏驱动中，通常在probe函数中会调用input\_set\_abs\_params给设备的input\_dev结构体初始化，这些input\_dev的参数会在Android的EventHub.cpp中被读取。如上可知，如果我们的触摸屏想被当成多点屏被处理，只需要在驱动中给input\_dev额外增加以下几个参数即可：

input\_set\_abs\_params（mcs\_data.input, ABS\_MT\_POSITION\_X, pdata->abs\_x\_min, pdata->abs\_x\_max, 0, 0）；

input\_set\_abs\_params（mcs\_data.input, ABS\_MT\_POSITION\_Y, pdata->abs\_y\_min, pdata->abs\_y\_max, 0, 0）；

input\_set\_abs\_params（mcs\_data.input, ABS\_MT\_TOUCH\_MAJOR, 0, 15, 0, 0）；

//相当于单点屏的ABX\_PRESSURE

input\_set\_abs\_params（mcs\_data.input, ABS\_MT\_WIDTH\_MAJOR, 0, 15, 0, 0）；

//相当于单点屏的ABS\_TOOL\_WIDTH

由于多点触摸技术需要采集到多个点，然后再一起处理这些点，所以在软件实现中需要保证每一波点的准确性和完整性。因此，Linux内核提供了input\_mt\_sync（struct input\_dev \* input）函数。在每波的每个点上报后需要紧跟一句input\_mt\_sync（）， 当这波所有点上报后再使用input\_sync（）进行同步。

例如一波要上报3个点：

……

input\_mt\_sync（input）；

……

input\_mt\_sync（input）；

……

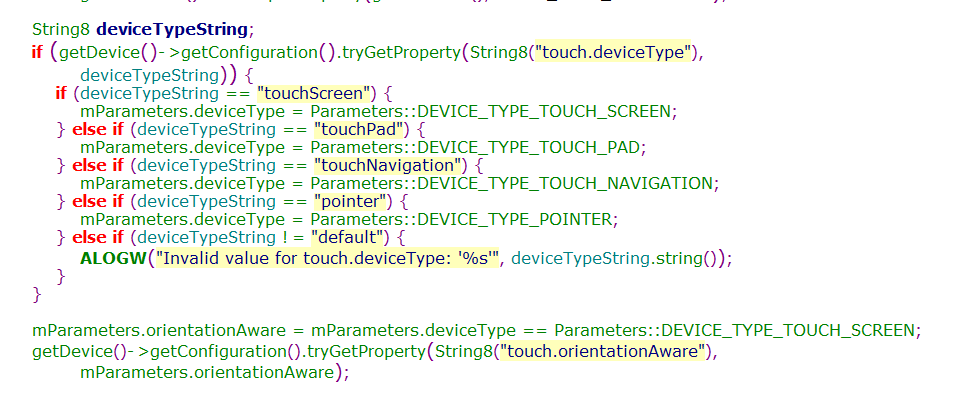
input\_mt\_sync（input）；

input\_sync（input）；

注：即使是仅上报一个点的单点事件，也需要一次input\_my\_sync。

touch.deviceType

InputReader.cpp (frameworks\native\services\inputflinger) 295464 2023/8/16



**触摸设备**

Android 支持多种触摸屏和触摸板，包括基于触控笔的数字化板。

触摸屏是与显示屏相关联的触摸设备，使用户能够在屏幕上直接操纵内容。

触摸板是不与显示屏相关联的触摸设备（如数字化板）。触摸板通常用于指控或绝对间接定位或基于手势的界面控制。

触摸设备可能具有功能与鼠标按钮类似的按钮。

有时可以使用各种不同的工具（如手指或触控笔）操作触摸设备，具体取决于底层的触摸传感器技术。

触摸设备有时用于实现虚拟按键。例如，在某些 Android 设备上，触摸屏传感器区域延伸超出显示屏的边缘，作为触摸式键盘的一部分发挥双重作用。

由于触摸设备种类繁多，因此 Android 依赖于大量配置属性来描述每种设备的特征和预期行为。

## 触摸设备分类

如果同时满足以下两个条件，那么输入设备属于多点触控设备：

输入设备报告存在 ABS\_MT\_POSITION\_X 和 ABS\_MT\_POSITION\_Y 绝对轴。

输入设备没有任何游戏手柄按钮。某些游戏手柄会使用与 MT 轴的代码重叠的代码来报告轴，而这一条件则消除了这种歧义。

如果同时满足以下两个条件，那么输入设备属于单点触控设备：

输入设备不属于多点触控设备。输入设备要么属于单点触控设备，要么属于多点触控设备，而不会同时属于这两种类别。

输入设备报告存在 ABS\_X 和 ABS\_Y 绝对轴以及 BTN\_TOUCH 按键代码。

如果输入设备属于触摸设备，系统会通过尝试加载设备的虚拟按键映射文件来确定是否存在虚拟按键。如果存在虚拟按键映射，则还会加载设备的按键布局文件。

有关虚拟按键映射文件的位置和格式，请参阅下面的部分。

接下来，系统会加载触摸设备的输入设备配置文件。

****所有内置触摸设备都应具有输入设备配置文件****。如果没有输入设备配置文件，系统将选择适用于典型通用触摸外围设备（如外部 USB 或蓝牙 HID 触摸屏或触摸板）的默认配置。这些默认配置不适用于内置触摸屏，很可能会导致错误的行为。

加载输入设备配置后，系统会将输入设备分类为触摸屏、触摸板或指控设备。

触摸屏设备用于直接操纵屏幕上的对象。由于用户直接触摸屏幕，因此系统不需要任何额外的感知性来指示被操纵的对象。

触摸板设备用于向应用提供关于在给定传感器区域上进行触摸时的绝对定位信息。它可能对数字化板有用。

指控设备用于使用光标间接操纵屏幕上的对象。手指被解释为多点触控指控手势。其他工具（如触控笔）则通过绝对位置来解释。

有关详情，请参阅[间接多点触控指控手势](https://source.android.com/docs/core/interaction/input/touch-devices?hl=zh-cn" \l "indirect-multi-touch-pointer-gestures)。

以下规则用于将输入设备归类为触摸屏、触摸板或指控设备。

如果设置了 touch.deviceType 属性，将按照指示设置设备类型。

如果输入设备报告存在 INPUT\_PROP\_DIRECT 输入属性（通过 EVIOCGPROP ioctl），设备类型将设置为触摸屏。该条件假设直接输入触摸设备已连接到同样处于连接状态的显示屏。

如果输入设备报告存在 INPUT\_PROP\_POINTER 输入属性（通过 EVIOCGPROP ioctl），设备类型将设置为指控设备。

如果输入设备报告存在 REL\_X 或 REL\_Y 相对轴，设备类型将设置为触摸板。该条件消除了由鼠标和触摸板组成的输入设备存在的歧义。在这种情况下，触摸板不会用于控制指针，因为指针已由鼠标控制。

否则，设备类型将被设置为指控设备。该默认设置确保没有指定任何其他特殊用途的触摸板将用于控制指控设备。

## 按钮

按钮是可供应用执行其他功能的“可选”控件。触摸设备上的按钮与鼠标按钮类似，主要与“指控式”触摸设备或者触控笔配合使用。

支持以下按钮：

BTN\_LEFT：映射到 MotionEvent.BUTTON\_PRIMARY。

BTN\_RIGHT：映射到 MotionEvent.BUTTON\_SECONDARY。

BTN\_MIDDLE：映射到 MotionEvent.BUTTON\_MIDDLE。

BTN\_BACK 和 BTN\_SIDE：映射到 MotionEvent.BUTTON\_BACK。按此按钮还可以合成按键（使用按键代码 KeyEvent.KEYCODE\_BACK）。

BTN\_FORWARD 和 BTN\_EXTRA：映射到 MotionEvent.BUTTON\_FORWARD。按此按钮还可以合成按键（使用按键代码 KeyEvent.KEYCODE\_FORWARD）。

BTN\_STYLUS：映射到 MotionEvent.BUTTON\_SECONDARY。

BTN\_STYLUS2：映射到 MotionEvent.BUTTON\_TERTIARY。

## 工具和工具类型

“工具”是指用于和触摸设备进行交互的手指、触控笔或其他装置。有些触摸设备可以区分不同类型的工具。

在 Android 的其他位置（和在 MotionEvent API 中一样），“工具”通常称为“指针”。

支持以下工具类型：

BTN\_TOOL\_FINGER 和 MT\_TOOL\_FINGER：映射到 MotionEvent.TOOL\_TYPE\_FINGER。

BTN\_TOOL\_PEN 和 MT\_TOOL\_PEN：映射到 MotionEvent.TOOL\_TYPE\_STYLUS。

BTN\_TOOL\_RUBBER：映射到 MotionEvent.TOOL\_TYPE\_ERASER。

BTN\_TOOL\_BRUSH：映射到 MotionEvent.TOOL\_TYPE\_STYLUS。

BTN\_TOOL\_PENCIL：映射到 MotionEvent.TOOL\_TYPE\_STYLUS。

BTN\_TOOL\_AIRBRUSH：映射到 MotionEvent.TOOL\_TYPE\_STYLUS。

BTN\_TOOL\_MOUSE：映射到 MotionEvent.TOOL\_TYPE\_MOUSE。

BTN\_TOOL\_LENS：映射到 MotionEvent.TOOL\_TYPE\_MOUSE。

BTN\_TOOL\_DOUBLETAP、BTN\_TOOL\_TRIPLETAP 和 BTN\_TOOL\_QUADTAP：映射到 MotionEvent.TOOL\_TYPE\_FINGER。

## 悬停与触摸工具

工具可以与触摸设备接触，也可以在触摸设备的感应范围内悬停在设备的上方。并非所有触摸设备都能够感应到悬停在其上方的工具。那些可实现感应的触摸设备（如基于射频的触控笔数字化仪）通常能在工具进入数字化仪有限的感应范围后检测到该工具。

InputReader 组件会谨慎区分触摸工具和悬停工具。同样，触摸工具和悬停工具也会以不同的方式报告给应用。

系统将使用 MotionEvent.ACTION\_DOWN、MotionEvent.ACTION\_MOVE、MotionEvent.ACTION\_DOWN、MotionEvent.ACTION\_POINTER\_DOWN 和 MotionEvent.ACTION\_POINTER\_UP 将触摸工具作为触摸事件报告给应用，

并使用 MotionEvent.ACTION\_HOVER\_ENTER、MotionEvent.ACTION\_HOVER\_MOVE 和 MotionEvent.ACTION\_HOVER\_EXIT 将悬停工具作为通用动作事件报告给应用。

## 触摸设备驱动程序要求

触摸设备驱动程序应该仅注册实际支持的轴/按钮的轴/按键代码。如果注册多余的轴/按键代码，可能会给设备分类算法带来混淆，或导致系统错误地检测设备的功能。

例如，如果设备报告 BTN\_TOUCH 按键代码，系统会假设 BTN\_TOUCH 将始终用于指示工具是否实际触摸到屏幕。因此，不应使用 BTN\_TOUCH 来指示该工具只是在感应范围内悬停。

单点触控设备使用以下 Linux 输入事件：

ABS\_X：（必需）报告工具的 X 坐标。

ABS\_Y：（必需）报告工具的 Y 坐标。

ABS\_PRESSURE：（可选）报告应用于工具尖端的物理压力或触摸接触面的信号强度。

ABS\_TOOL\_WIDTH：（可选）报告触摸接触面或工具本身的横截面积或宽度。

ABS\_DISTANCE：（可选）报告工具与触摸设备表面之间的距离。

ABS\_TILT\_X：（可选）报告工具沿触摸设备表面 X 轴方向的倾斜度。

ABS\_TILT\_Y：（可选）报告工具沿触摸设备表面 Y 轴方向的倾斜度。

BTN\_TOUCH：（必需）指示工具是否触摸到设备。

BTN\_LEFT、BTN\_RIGHT、BTN\_MIDDLE、BTN\_BACK、BTN\_SIDE、BTN\_FORWARD、BTN\_EXTRA、BTN\_STYLUS、BTN\_STYLUS2：（可选）报告[按钮](https://source.android.com/docs/core/interaction/input/touch-devices?hl=zh-cn" \l "buttons)状态。

BTN\_TOOL\_FINGER、BTN\_TOOL\_PEN、BTN\_TOOL\_RUBBER、BTN\_TOOL\_BRUSH、BTN\_TOOL\_PENCIL、BTN\_TOOL\_AIRBRUSH、BTN\_TOOL\_MOUSE、BTN\_TOOL\_LENS、BTN\_TOOL\_DOUBLETAP、BTN\_TOOL\_TRIPLETAP、BTN\_TOOL\_QUADTAP：（可选）报告[工具类型](https://source.android.com/docs/core/interaction/input/touch-devices?hl=zh-cn" \l "tools-and-tool-types)。

多点触控设备使用以下 Linux 输入事件：

ABS\_MT\_POSITION\_X：（必需）报告工具的 X 坐标。

ABS\_MT\_POSITION\_Y：（必需）报告工具的 Y 坐标。

ABS\_MT\_PRESSURE：（可选）报告应用于工具尖端的物理压力或触摸接触面的信号强度。

ABS\_MT\_TOUCH\_MAJOR：（可选）报告触摸接触面的横截面积或触摸接触面较长尺寸的长度。

ABS\_MT\_TOUCH\_MINOR：（可选）报告触摸区域较短尺寸的长度。如果 ABS\_MT\_TOUCH\_MAJOR 报告区域测量，不应使用此轴。

ABS\_MT\_WIDTH\_MAJOR：（可选）报告工具本身的横截面积或工具本身较长方向的长度。如果工具本身的尺寸未知，不应使用此轴。

ABS\_MT\_WIDTH\_MINOR：（可选）报告工具本身较短尺寸的长度。如果 ABS\_MT\_WIDTH\_MAJOR 报告区域测量尺寸，或者如果工具本身的尺寸未知，则不应使用此轴。

ABS\_MT\_ORIENTATION：（可选）报告工具的方向。

ABS\_MT\_DISTANCE：（可选）报告工具与触摸设备表面之间的距离。

ABS\_MT\_TOOL\_TYPE：（可选）将[工具类型](https://source.android.com/docs/core/interaction/input/touch-devices?hl=zh-cn" \l "tools-and-tool-types)报告为 MT\_TOOL\_FINGER 或 MT\_TOOL\_PEN。

ABS\_MT\_TRACKING\_ID：（可选）报告工具的跟踪 ID。跟踪 ID 是一个任意的非负整数。当多个工具同时处于活动状态时，可以使用该 ID 独立识别和跟踪各个工具。例如，当多个手指同时触摸设备时，应为每个手指分配一个不同的跟踪 ID，用于在手指触摸设备期间识别手指。跟踪 ID 可在其关联的工具移出感应范围后重复使用。

ABS\_MT\_SLOT：（可选）在使用 Linux 多点触控协议“B”时，报告工具的槽位 ID。有关详情，请参阅 Linux 多点触控协议文档。

BTN\_TOUCH：（必需）指示工具是否触摸到设备。

BTN\_LEFT、BTN\_RIGHT、BTN\_MIDDLE、BTN\_BACK、BTN\_SIDE、BTN\_FORWARD、BTN\_EXTRA、BTN\_STYLUS、BTN\_STYLUS2：（可选）报告[按钮](https://source.android.com/docs/core/interaction/input/touch-devices?hl=zh-cn" \l "buttons)状态。

BTN\_TOOL\_FINGER、BTN\_TOOL\_PEN、BTN\_TOOL\_RUBBER、BTN\_TOOL\_BRUSH、BTN\_TOOL\_PENCIL、BTN\_TOOL\_AIRBRUSH、BTN\_TOOL\_MOUSE、BTN\_TOOL\_LENS、BTN\_TOOL\_DOUBLETAP、BTN\_TOOL\_TRIPLETAP、BTN\_TOOL\_QUADTAP：（可选）报告[工具类型](https://source.android.com/docs/core/interaction/input/touch-devices?hl=zh-cn" \l "tools-and-tool-types)。

如果同时针对单点触控协议和多点触控协议定义了轴，则仅会使用多点触控轴，并忽略单点触控轴。

ABS\_X、ABS\_Y、ABS\_MT\_POSITION\_X 和 ABS\_MT\_POSITION\_Y 轴的最小值和最大值用于指定设备有效区域的范围，采用适用于具体设备的 surface 单位。如果是触摸屏，有效区域是指触摸设备实际覆盖显示屏的部分。

对于触摸屏，系统会自动插入报告的触摸位置（采用 surface 单位），以通过以下公式计算得出以显示像素表示的触摸位置：

displayX = (x - minX) \* displayWidth / (maxX - minX + 1)

displayY = (y - minY) \* displayHeight / (maxY - minY + 1)

触摸屏可能会报告在报告的有效区域之外发起的触摸。

在有效区域之外发起的触摸不会传递给应用，但可用于虚拟按键。

在有效区域内发起的触摸或进入和退出显示区域的触摸会传递给应用。因此，如果触摸是在应用的范围内开始，然后移动到有效区域之外，那么应用可能会收到显示坐标为负或超出显示范围的触摸事件。这属于正常现象。

触摸设备不得限制有效区域的触摸坐标边界。如果触摸离开有效区域，则应将其报告为超出有效区域，或者根本不应报告。

例如，如果用户的手指在触摸屏左上角附近触摸，可能会报告 (minX, minY) 坐标。如果手指继续移动到有效区域之外，触摸屏应该开始报告分量小于 minX 和 minY 的坐标（如 (minX - 2, minY - 3)），或者完全停止报告触摸。换而言之，当用户的手指确实触摸到有效区域之外时，触摸屏不应该报告 (minX, minY)。

如果将触摸坐标限制为不超过显示屏边缘，会导致屏幕边缘周围形成人为硬边界，从而使系统无法顺畅地跟踪进入或离开显示区域边界的动作。

ABS\_PRESSURE 或 ABS\_MT\_PRESSURE 报告的值（如果有报告）在工具触摸设备时必须为非零值；否则就为零，表示该工具处于悬停状态。

可以选择性报告压力信息，但强烈建议报告。应用可以使用压力信息实现压敏绘图等效果。

ABS\_TOOL\_WIDTH、ABS\_MT\_TOUCH\_MAJOR、ABS\_MT\_TOUCH\_MINOR、ABS\_MT\_WIDTH\_MAJOR 或 ABS\_MT\_WIDTH\_MINOR 报告的值在工具触摸设备时应为非零值；否则就为零，但这不是必需的。例如，触摸设备可能能够测量手指触摸接触面的尺寸，但不能测量触控笔触摸接触面的尺寸。

可以选择性报告大小信息，但强烈建议报告。应用可以使用压力信息来实现尺寸敏感绘图等效果。

在工具触摸设备时，ABS\_DISTANCE 或 ABS\_MT\_DISTANCE 报告的值应接近零。即使当工具处于直接接触时，距离仍可能为非零。报告的确切值取决于硬件测量距离的方式。

可以自行选择是否报告距离信息，但对于触控笔设备，建议报告该信息。

当工具垂直于设备时，ABS\_TILT\_X 和 ABS\_TILT\_Y 报告的值应为零。将非零倾斜作为工具保持在倾斜处的标志。

假定沿 X 轴和 Y 轴的倾斜角度以与垂直方向的夹角计。中心点（完全垂直）由 (max + min) / 2 针对每个轴指定。小于中心点的值表示向上或向左倾斜，大于中心点的值表示向下或向右倾斜。

InputReader 将 X 和 Y 倾斜分量转换成从 0 到 PI / 2 弧度的垂直倾斜角以及从 -PI 到 PI 弧度的平面定向角。该表示法将产生与描述手指触摸所用方向相符的方向描述。

可以自行选择是否报告倾斜信息，但对于触控笔设备，建议报告该信息。

如果工具类型是由 ABS\_MT\_TOOL\_TYPE 报告的，它会取代 BTN\_TOOL\_\* 报告的任何工具类型信息。如果没有任何工具类型信息，工具类型将默认为 MotionEvent.TOOL\_TYPE\_FINGER。

根据以下条件确定工具的活动状态：

当使用单点触控协议时，如果 BTN\_TOUCH 或 BTN\_TOOL\_\* 为 1，表示工具处于活动状态。

这个条件意味着 InputReader 至少需要获得一些关于工具性质的信息：工具是否正在触摸屏幕，或者至少获取工具的类型。如果没有可用的信息，它会假定工具处于非活动状态（超出范围）。

当使用多点触控协议“A”时，只要工具出现在最近的同步报告中，则表示其处于活动状态。当工具不再出现在同步报告中时，则表示工具已不再处于活动状态。

当使用多点触控协议“B”时，只要工具具有活动槽位，则表示其处于活动状态。当插槽被清除时，则表示工具不再存在。

根据以下条件确定工具悬停：

如果工具为 BTN\_TOOL\_MOUSE 或 BTN\_TOOL\_LENS，则工具不会悬停，即使以下任一条件为 true 也不例外。

如果工具处于活动状态，并且驱动程序报告的压力为零，则表示工具处于悬停状态。

如果工具处于活动状态，驱动程序支持 BTN\_TOUCH 按键代码，并且 BTN\_TOUCH 的值为零，则表示工具处于悬停状态。

InputReader 支持多点触控协议“A”和“B”。新驱动程序应该使用“B”协议，但是使用任一协议均可正常运作。

****从 Android Ice Cream Sandwich 4.0 开始，可能需要更改触摸屏驱动程序，以符合 Linux 输入协议规范。****

可能需要进行以下更改：

当一个工具变为非活动状态（手指“抬起”）时，它应该停止显示在后续的多点触控同步报告中。当所有工具变为非活动状态（“抬起”所有手指）时，驱动程序应发送一个空的同步报告数据包，如 SYN\_MT\_REPORT 后跟 SYN\_REPORT。

以前版本的 Android 通过发送压力值 0 来报告“抬起”事件。这种旧行为不符合 Linux 输入协议规范，因此不再受支持。

物理压力或信号强度信息应使用 ABS\_MT\_PRESSURE 进行报告。

以前版本的 Android 从 ABS\_MT\_TOUCH\_MAJOR 检索压力信息。这种旧行为不符合 Linux 输入协议规范，因此不再受支持。

触摸尺寸信息应使用 ABS\_MT\_TOUCH\_MAJOR 进行报告。

以前版本的 Android 从 ABS\_MT\_TOOL\_MAJOR 检索尺寸信息。这种旧行为不符合 Linux 输入协议规范，因此不再受支持。

触摸设备驱动程序不再需要 Android 系统专用的自定义设置。通过依靠标准的 Linux 输入协议，Android 可以使用未经修改的驱动程序支持更多种类的触摸外围设备，如外部 HID 多点触控触摸屏。

## 触摸设备操作

下面简要汇总了 Android 上的触摸设备操作。

EventHub 从 evdev 驱动程序读取原始事件。

InputReader 会使用原始事件，并更新关于每个工具的位置和其他特征的内部状态。它还会跟踪按钮状态。

如果按下或释放“后退”或“前进”按钮，InputReader 会向 InputDispatcher 发出按键事件通知。

InputReader 确定是否发生了虚拟按键的按压操作。如果是，它会向 InputDispatcher 发出按键事件通知。

InputReader 确定触摸行为是否是在显示范围内发起的。如果是，它会向 InputDispatcher 发出触摸事件通知。

如果没有触摸工具，但至少有一个悬停工具，那么 InputReader 会向 InputDispatcher 发出悬停事件通知。

如果触摸设备类型是指控设备，InputReader 会执行指控手势检测，相应地移动指针和相关点，并向 InputDispatcher 发出指针事件通知。

InputDispatcher 会根据 WindowManagerPolicy 确定是否应分派这些事件，以及它们是否应唤醒设备。然后，InputDispatcher 会将事件传递给相应的应用。

## 触摸设备配置

触摸设备行为由设备的坐标轴、按钮、输入属性、输入设备配置、虚拟按键映射和按键布局确定。

如需详细了解参与键盘配置的文件，请参阅以下部分：

* [输入设备配置文件](https://source.android.com/docs/core/interaction/input/input-device-configuration-files?hl=zh-cn)
* [虚拟按键映射文件](https://source.android.com/docs/core/interaction/input/touch-devices?hl=zh-cn" \l "virtual-key-map-files)

### 属性

系统依赖于许多输入设备配置属性来配置和校准触摸设备行为。

原因之一是触摸设备的设备驱动程序通常使用特定于设备的单元来报告触摸特性。

例如，许多触摸设备使用内部特定于设备的比例（例如由触摸触发的传感器节点的总数）来测量触摸接触面积。此原始尺寸值对应用来说没有意义，因为它们需要了解触摸设备传感器节点的物理尺寸和其他特性。

系统会使用在输入设备配置文件中编码的校准参数将触摸设备报告的值解码、转换，并标准化为应用可以理解的更简单的标准表示形式。

### 文档规范

对本文档而言，我们将使用以下规范来描述系统在校准过程中使用的值。

#### 原始轴值

以下表达式表示触摸设备驱动程序作为 EV\_ABS 事件报告的原始值。

raw.x

ABS\_X 或 ABS\_MT\_POSITION\_X 轴的值。

raw.y

ABS\_Y 或 ABS\_MT\_POSITION\_Y 轴的值。

raw.pressure

ABS\_PRESSURE 或 ABS\_MT\_PRESSURE 轴的值，如果未提供，则为 0。

raw.touchMajor

ABS\_MT\_TOUCH\_MAJOR 轴的值，如果未提供，则为 0。

raw.touchMinor

ABS\_MT\_TOUCH\_MINOR 轴的值，如果未提供，则为 raw.touchMajor。

raw.toolMajor

ABS\_TOOL\_WIDTH 或 ABS\_MT\_WIDTH\_MAJOR 轴的值，如果未提供，则为 0。

raw.toolMinor

ABS\_MT\_WIDTH\_MINOR 轴的值，如果未提供，则为 raw.toolMajor。

raw.orientation

ABS\_MT\_ORIENTATION 轴的值，如果未提供，则为 0。

raw.distance

ABS\_DISTANCE 或 ABS\_MT\_DISTANCE 轴的值，如果未提供，则为 0。

raw.tiltX

ABS\_TILT\_X 轴的值，如果未提供，则为 0。

raw.tiltY

ABS\_TILT\_Y 轴的值，如果未提供，则为 0。

#### 原始轴范围

以下表达式表示原始值的范围。通过为每个轴调用 EVIOCGABS ioctl 获取它们。

raw.\*.min

原始轴的最小值（含）。

raw.\*.max

原始轴的最大值（含）。

raw.\*.range

相当于 raw.\*.max - raw.\*.min。

raw.\*.fuzz

原始轴的精度。例如，fuzz = 1 表示值精确到 +/- 1 个单位。

raw.width

触摸区域的宽度（含），相当于 raw.x.range + 1。

raw.height

触摸区域的高度（含），相当于 raw.y.range + 1。

#### 输出范围

以下表达式表示输出坐标系的特性。系统使用线性插值将触摸设备使用的 Surface 单元的触摸位置信息转换成将报告给应用的输出单元（如显示像素）。

output.width

输出宽度。对于触摸屏（与显示屏相关联），输出宽度是显示屏宽度（以像素为单位）。对于触摸板（不与显示屏相关联），输出宽度等于 raw.width，表示不会执行任何插值。

output.height

输出高度。对于触摸屏（与显示屏相关联），输出高度是显示屏高度（以像素为单位）。对于触摸板（不与显示屏相关联），输出高度等于 raw.height，表示不会执行任何插值。

output.diag

输出坐标系的对角线长度，相当于 sqrt(output.width ^2 + output.height ^2)。

### 基本配置

触摸输入映射器在输入设备配置文件中使用许多配置属性来指定校准值。下表介绍了一些通用配置属性。下面的部分中介绍了所有其他属性以及使用这些属性进行校准时所用的字段。

#### touch.deviceType

定义：touch.deviceType = touchScreen | touchPad | pointer | default

指定触摸设备类型。

如果值为 touchScreen，那么触摸设备是与显示屏相关联的触摸屏。

如果值为 touchPad，那么触摸设备是不与显示屏相关联的触摸板。

如果值为 pointer，那么触摸设备是不与显示屏相关联的触摸板，并且其动作用于[间接多点触控指控手势](https://source.android.com/docs/core/interaction/input/touch-devices?hl=zh-cn" \l "indirect-multi-touch-pointer-gestures)。

如果值为 default，系统将根据分类算法自动检测设备类型。

有关设备类型如何影响触摸设备的行为的详细信息，请参阅[分类](https://source.android.com/docs/core/interaction/input/touch-devices?hl=zh-cn" \l "touch-device-classification)部分。

在 Honeycomb 之前，所有触摸设备都被视为触摸屏。

#### touch.orientationAware

定义：touch.orientationAware = 0 | 1

指定触摸设备是否应对显示屏的方向更改做出响应。

如果值为 1，只要显示屏的方向更改了，触摸设备报告的触摸位置就会旋转。

如果值为 0，触摸设备报告的触摸位置将不受显示屏方向更改的影响。

如果设备是触摸屏，默认值为 1，否则为 0。

系统会区分内部和外部触摸屏与显示部分。方向感知型内部触摸屏基于内部显示部分的方向进行旋转。方向感知型外部触摸屏基于外部显示部分的方向进行旋转。

方向感知功能用于支持 Nexus One 等设备上的触摸屏旋转。例如，当设备从其自然方向顺时针旋转 90 度时，触摸的绝对位置将被重新映射，使得在触摸屏绝对坐标系左上角的触摸行为被报告为在显示屏旋转后坐标系左上角的触摸行为。这样做是为了使用应用绘制其可见元素时所用的同一坐标系报告触摸行为。

在 Honeycomb 之前，所有触摸设备都被视为具有方向感知功能。

#### touch.gestureMode

定义：touch.gestureMode = pointer | spots | default

指定指控手势的表示模式。仅在触摸设备为指控类型时，该配置属性才具有相关性。

如果值为 pointer，触摸板手势将通过与鼠标指针类似的光标来表示。

如果值为 spots，触摸板手势将通过代表手势形心的锚点和代表各个手指位置的一组圆形斑点来表示。

如果设置了 INPUT\_PROP\_SEMI\_MT 输入属性，默认值为 pointer，否则为 spots。

### X 和 Y 字段

X 和 Y 字段给出了接触区域中心的位置信息。

#### 计算

计算非常简单：以线性方式将触摸驱动程序的位置信息插入输出坐标系。

xScale = output.width / raw.width

yScale = output.height / raw.height

If not orientation aware or screen rotation is 0 degrees:

output.x = (raw.x - raw.x.min) \* xScale

output.y = (raw.y - raw.y.min) \* yScale

Else If rotation is 90 degrees:

output.x = (raw.y - raw.y.min) \* yScale

output.y = (raw.x.max - raw.x) \* xScale

Else If rotation is 180 degrees:

output.x = (raw.x.max - raw.x) \* xScale

output.y = (raw.y.max - raw.y) \* yScale

Else If rotation is 270 degrees:

output.x = (raw.y.max - raw.y) \* yScale

output.y = (raw.x - raw.x.min) \* xScale

End If

### TouchMajor、TouchMinor、ToolMajor、ToolMinor、Size 字段

TouchMajor 和 TouchMinor 字段用于描述接触面的大致尺寸，采用输出单位（像素）。

ToolMajor 和 ToolMinor 字段用于描述[工具](https://source.android.com/docs/core/interaction/input/touch-devices?hl=zh-cn" \l "tools-and-tool-types)本身的大致尺寸，采用输出单位（像素）。

Size 字段描述了相对于触摸设备可以感知的最大可能触摸区域的标准化触摸区域尺寸。可能的最小标准化尺寸为 0.0（无接触或不可测量），可能的最大标准化尺寸为 1.0（传感器区域已经完全覆盖）。

如果可以同时测量近似长度和宽度，TouchMajor 字段会指定接触区域的较长尺寸，TouchMinor 字段会指定接触区域的较短尺寸。如果只能测量接触区域的大致直径，TouchMajor 和 TouchMinor 字段将相等。

同样，ToolMajor 字段用于指定工具横截面较长方向的尺寸，ToolMinor 字段用于指定工具横截面较短方向的尺寸。

如果触摸尺寸不可测量，但工具尺寸可测量，那么工具尺寸将设为等于触摸尺寸。相反，如果工具尺寸不可测量，但触摸尺寸可测量，那么触摸尺寸将设为等于工具尺寸。

触摸设备以多种方式测量或报告触摸尺寸和工具尺寸。目前的实现支持三种不同的测量尺寸：直径、面积以及采用 surface 单位表示的几何边界框。

#### touch.size.calibration

定义：touch.size.calibration = none | geometric | diameter | area | default

指定触摸驱动程序报告触摸尺寸和工具尺寸时所用的测量类型。

如果值为 none，那么尺寸设为零。

如果值为 geometric，那么假定以与位置相同的 Surface 单元指定尺寸，从而以相同的方式对尺寸进行缩放。

如果值为 diameter，那么假定尺寸与触摸或工具直径（宽度）成比例。

如果值为 area，那么假定尺寸与触摸或工具面积成比例。

如果值为 default，那么在 raw.touchMajor 或 raw.toolMajor 轴可用的情况下，系统将使用 geometric 校准，否则使用 none 校准。

#### touch.size.scale

定义：touch.size.scale = <非负浮点数>

指定进行校准时使用的恒定缩放比例。

默认值为 1.0。

#### touch.size.bias

定义：touch.size.bias = <非负浮点数>

指定校准中使用的恒定偏差值。

默认值为 0.0。

#### touch.size.isSummed

定义：touch.size.isSummed = 0 | 1

指定尺寸是报告为所有有效接触区域的尺寸总和，还是针对每个接触区域单独报告尺寸。

如果值为 1，报告的尺寸需除以接触区域数量，然后才能使用。

如果值为 0，报告的尺寸将按原样使用。

默认值为 0。

一些触摸设备（尤其是“Semi-MT”设备）无法区分多个接触面的单独尺寸，因此它们会报告表示总面积或总宽度的尺寸测量结果。对于此类设备，此属性只能设为 1。如果不确定，请将此值设为 0。

#### 计算

TouchMajor、TouchMinor、ToolMajor、ToolMinor 和 Size 字段的计算方法取决于指定的校准参数。

If raw.touchMajor and raw.toolMajor are available:

touchMajor = raw.touchMajor

touchMinor = raw.touchMinor

toolMajor = raw.toolMajor

toolMinor = raw.toolMinor

Else If raw.touchMajor is available:

toolMajor = touchMajor = raw.touchMajor

toolMinor = touchMinor = raw.touchMinor

Else If raw.toolMajor is available:

touchMajor = toolMajor = raw.toolMajor

touchMinor = toolMinor = raw.toolMinor

Else

touchMajor = toolMajor = 0

touchMinor = toolMinor = 0

size = 0

End If

size = avg(touchMajor, touchMinor)

If touch.size.isSummed == 1:

touchMajor = touchMajor / numberOfActiveContacts

touchMinor = touchMinor / numberOfActiveContacts

toolMajor = toolMajor / numberOfActiveContacts

toolMinor = toolMinor / numberOfActiveContacts

size = size / numberOfActiveContacts

End If

If touch.size.calibration == "none":

touchMajor = toolMajor = 0

touchMinor = toolMinor = 0

size = 0

Else If touch.size.calibration == "geometric":

outputScale = average(output.width / raw.width, output.height / raw.height)

touchMajor = touchMajor \* outputScale

touchMinor = touchMinor \* outputScale

toolMajor = toolMajor \* outputScale

toolMinor = toolMinor \* outputScale

Else If touch.size.calibration == "area":

touchMajor = sqrt(touchMajor)

touchMinor = touchMajor

toolMajor = sqrt(toolMajor)

toolMinor = toolMajor

Else If touch.size.calibration == "diameter":

touchMinor = touchMajor

toolMinor = toolMajor

End If

If touchMajor != 0:

output.touchMajor = touchMajor \* touch.size.scale + touch.size.bias

Else

output.touchMajor = 0

End If

If touchMinor != 0:

output.touchMinor = touchMinor \* touch.size.scale + touch.size.bias

Else

output.touchMinor = 0

End If

If toolMajor != 0:

output.toolMajor = toolMajor \* touch.size.scale + touch.size.bias

Else

output.toolMajor = 0

End If

If toolMinor != 0:

output.toolMinor = toolMinor \* touch.size.scale + touch.size.bias

Else

output.toolMinor = 0

End If

output.size = size

### Pressure 字段

Pressure 字段用于描述施加到触摸设备的大概物理压力，以介于 0.0（无接触）和 1.0（全力）之间的标准化值形式表示。

零压力表示工具处于悬停状态。

#### touch.pressure.calibration

定义：touch.pressure.calibration = none | physical | amplitude | default

指定触摸驱动程序报告压力所用的测量类型。

如果值为 none，压力未知，那么在触摸时设置为 1.0，悬停时设置为 0.0。

如果值为 physical，那么认为压力轴测量的是施加到触摸板的压力的实际物理强度。

如果值为 amplitude，那么认为压力轴测量的是信号幅度（与接触面的大小和施加的压力有关）。

如果值为 default，在压力轴可用的情况下，系统将使用 physical 校准，否则使用 none。

#### touch.pressure.scale

定义：touch.pressure.scale = <非负浮点数>

指定进行校准时使用的恒定缩放比例。

默认值为 1.0 / raw.pressure.max。

#### 计算

Pressure 字段的计算方法取决于指定的校准参数。

If touch.pressure.calibration == "physical" or "amplitude":

output.pressure = raw.pressure \* touch.pressure.scale

Else

If hovering:

output.pressure = 0

Else

output.pressure = 1

End If

End If

### Orientation 和 Tilt 字段

Orientation 字段以角度测量的形式描述触摸和工具的方向。值为 0 表示长轴为垂直方向，-PI/2 表示长轴朝向左方，PI/2 表示长轴朝向右方。当存在触控笔工具时，方向范围可以是从 -PI 到 PI 的整个圆环范围。

Tilt 字段通过测量角度描述工具的倾斜度。倾斜度为 0 表示工具垂直于表面。倾斜度为 PI/2 表示工具与表面平行。

#### touch.orientation.calibration

定义：touch.orientation.calibration = none | interpolated | vector | default

指定触摸驱动程序报告方向时所用的测量类型。

如果值为 none，方向未知，则设为 0。

如果值为 interpolated，方向会被线性插入，使得 raw.orientation.min 的原始值映射到 -PI/2，raw.orientation.max 的原始值映射到 PI/2。(raw.orientation.min + raw.orientation.max) / 2 的中心值映射到 0。

如果值为 vector，则方向表示为包含两个带符号的 4 位字段的压缩向量。该表示法用于 Atmel 基于对象的协议部分。当解码时，向量生成定向角和置信度。置信度用于缩放尺寸信息，除非它是几何图形。

如果值为 default，那么在方向轴可用的情况下，系统将使用 interpolated 校准，否则使用 none。

#### 计算

Orientation 和 Tilt 字段的计算方法取决于指定的校准参数和可用输入。

If touch.tiltX and touch.tiltY are available:

tiltXCenter = average(raw.tiltX.min, raw.tiltX.max)

tiltYCenter = average(raw.tiltY.min, raw.tiltY.max)

tiltXAngle = (raw.tiltX - tiltXCenter) \* PI / 180

tiltYAngle = (raw.tiltY - tiltYCenter) \* PI / 180

output.orientation = atan2(-sin(tiltXAngle), sinf(tiltYAngle))

output.tilt = acos(cos(tiltXAngle) \* cos(tiltYAngle))

Else If touch.orientation.calibration == "interpolated":

center = average(raw.orientation.min, raw.orientation.max)

output.orientation = PI / (raw.orientation.max - raw.orientation.min)

output.tilt = 0

Else If touch.orientation.calibration == "vector":

c1 = (raw.orientation & 0xF0) >> 4

c2 = raw.orientation & 0x0F

If c1 != 0 or c2 != 0:

If c1 >= 8 Then c1 = c1 - 16

If c2 >= 8 Then c2 = c2 - 16

angle = atan2(c1, c2) / 2

confidence = sqrt(c1\*c1 + c2\*c2)

output.orientation = angle

If touch.size.calibration == "diameter" or "area":

scale = 1.0 + confidence / 16

output.touchMajor \*= scale

output.touchMinor /= scale

output.toolMajor \*= scale

output.toolMinor /= scale

End If

Else

output.orientation = 0

End If

output.tilt = 0

Else

output.orientation = 0

output.tilt = 0

End If

If orientation aware:

If screen rotation is 90 degrees:

output.orientation = output.orientation - PI / 2

Else If screen rotation is 270 degrees:

output.orientation = output.orientation + PI / 2

End If

End If

### Distance 字段

Distance 字段描述了工具和触摸设备表面之间的距离。值为 0.0 表示直接接触，值越大，表示与表面之间的距离越远。

#### touch.distance.calibration

定义：touch.distance.calibration = none | scaled | default

指定触摸驱动程序报告距离时所用的测量类型。

如果值为 none，距离未知，则设为 0。

如果值为 scaled，报告的距离将乘以恒定缩放比例。

如果值为 default，在距离轴可用的情况下，系统将使用 scaled 校准，否则使用 none。

#### touch.distance.scale

定义：touch.distance.scale = <非负浮点数>

指定进行校准时使用的恒定缩放比例。

默认值为 1.0。

#### 计算

Distance 字段的计算方法取决于指定的校准参数。

If touch.distance.calibration == "scaled":

output.distance = raw.distance \* touch.distance.scale

Else

output.distance = 0

End If

### 示例

# Input device configuration file for a touch screen that supports pressure,

# size and orientation. The pressure and size scale factors were obtained

# by measuring the characteristics of the device itself and deriving

# useful approximations based on the resolution of the touch sensor and the

# display.

#

# Note that these parameters are specific to a particular device model.

# Different parameters will need to be used for other devices.

# Basic Parameters

touch.deviceType = touchScreen

touch.orientationAware = 1

# Size

# Based on empirical measurements, we estimate the size of the contact

# using size = sqrt(area) \* 28 + 0.

touch.size.calibration = area

touch.size.scale = 28

touch.size.bias = 0

touch.size.isSummed = 0

# Pressure

# Driver reports signal strength as pressure.

#

# A normal index finger touch typically registers about 80 signal strength

# units although we don't expect these values to be accurate.

touch.pressure.calibration = amplitude

touch.pressure.scale = 0.0125

# Orientation

touch.orientation.calibration = vector

### 兼容性说明

触摸设备的配置属性在 Android Ice Cream Sandwich 4.0 中发生了重大变化。****必须更新触摸设备的所有输入设备配置文件，才能使用新的配置属性****。

更旧的触摸设备[驱动程序](https://source.android.com/docs/core/interaction/input/touch-devices?hl=zh-cn" \l "touch-device-driver-requirements)可能也需要更新。

## 虚拟按键映射文件

触摸设备经常用于实现虚拟按键。

有几种方法可以做到这一点，具体取决于触摸控制器的功能。一些触摸控制器可以直接配置为通过设置固件寄存器来实现软键。其他时候，最好在软件中执行从触摸坐标到按键代码的映射。

在软件中实现虚拟按键时，内核必须将名为 virtualkeys.<devicename> 的虚拟按键映射文件作为本机已加载属性导出。例如，如果触摸屏设备驱动程序将其名称报告为“touchyfeely”，虚拟按键映射文件的路径必须为 /sys/board\_properties/virtualkeys.touchyfeely。

虚拟按键映射文件描述了触摸屏上虚拟按键的坐标和 Linux 按键代码。

除了虚拟按键映射文件外，还必须有一个对应的按键布局文件和按键字符映射文件，以将 Linux 按键代码映射到 Android 按键代码，并指定键盘设备的类型（通常为 SPECIAL\_FUNCTION）。

### 语法

虚拟按键映射文件是一个纯文本文件，由一系列用换行符或冒号分隔的虚拟按键布局描述组成。

注释行以“#”开头，并持续到这一行的结束位置。

每个虚拟按键用由 6 个冒号分隔的组件进行描述：

* 0x01：版本代码。必须始终为 0x01。
* <Linux key code>：虚拟按键的 Linux 按键代码。
* <centerX>：虚拟按键中心的 X 轴坐标（以像素为单位）。
* <centerY>：虚拟按键中心的 Y 轴坐标（以像素为单位）。
* <width>：虚拟按键的宽度（以像素为单位）。
* <height>：虚拟按键的高度（以像素为单位）。

所有的坐标和尺寸都是根据显示坐标系指定的。

下面是一个虚拟按键映射文件，全部写在一行上。

# All on one line

0x01:158:55:835:90:55:0x01:139:172:835:125:55:0x01:102:298:835:115:55:0x01:217:412:835:95:55

相同的虚拟按键映射文件也可以写在多行上。

# One key per line

0x01:158:55:835:90:55

0x01:139:172:835:125:55

0x01:102:298:835:115:55

0x01:217:412:835:95:55

在上述示例中，触摸屏具有 480×800 的分辨率。因此，所有虚拟按键的 <centerY> 坐标均为 835，位于略低于触摸屏可见区域的位置。

第一个按键的 Linux 扫描代码为 158 (KEY\_BACK)，centerX 为 55，centerY 为 835，width 为 90，height 为 55。

### 示例

虚拟按键映射文件：/sys/board\_properties/virtualkeys.touchyfeely。

0x01:158:55:835:90:55

0x01:139:172:835:125:55

0x01:102:298:835:115:55

0x01:217:412:835:95:55

按键布局文件：/system/usr/keylayout/touchyfeely.kl。

key 158 BACK

key 139 MENU

key 172 HOME

key 217 SEARCH

按键字符映射文件：/system/usr/keychars/touchyfeely.kcm。

type SPECIAL\_FUNCTION

## 间接多点触控指控手势

在指控模式下，系统会解释以下手势：

单指点按：点击。

单指移动：移动指针。

单指移动加按下按钮：拖动指针。

两个手指移动（两个手指沿相同的方向移动）：沿着该方向拖动指针下方的区域。指针本身不动。

两个手指移动（两个手指朝着彼此移动或者移向不同方向）：平移/缩放/旋转指针周围的区域。指针本身不动。

多个手指移动：自由手势。

## 防手掌误触

从 Android 13 开始，系统可以在内置框架启用时自动拒绝来自手掌的输入。内部构建的自定义解决方案仍受支持，但可能需要进行修改，以便在检测到手掌时返回 TOOL\_TYPE\_PALM 标志。内置框架还可与自定义解决方案搭配使用。

实际模型会查看当前指针和周围指针处前 90 毫秒的手势数据，然后考虑触摸与屏幕边缘之间的距离。然后，它会基于每个指控设备确定哪些指控设备是手掌。它还会考虑每个接触面的大小（如 TouchMajor 和 TouchMinor 所报告）。然后，Android 框架会从触摸流中移除标记为手掌的指针。

如果指针已经发送到应用，系统会执行以下操作之一：

* （如果有其他处于活动状态的指针）使用 ACTION\_POINTER\_UP 和 FLAG\_CANCELED 组合取消该指针。
* （如果这是唯一的指针）使用 ACTION\_CANCEL 取消该指针。

公共 API MotionEvent.FLAG\_CANCELED 表示当前事件不应触发用户操作。此标志针对 ACTION\_CANCEL 和 ACTION\_POINTER\_UP 设置。

如果手掌指针没有发送到应用，系统会直接丢弃该指针。

### 启用防手掌误触功能

1. 在触摸驱动程序中，使用 [input\_abs\_set\_res 宏](https://elixir.bootlin.com/linux/latest/source/include/linux/input.h" \l "L486)设置下列字段的分辨率（单位为[每毫米像素数](https://github.com/torvalds/linux/blob/5bfc75d92efd494db37f5c4c173d3639d4772966/include/uapi/linux/input.h" \l "L81)）：
   * ABS\_MT\_POSITION\_X
   * ABS\_MT\_POSITION\_Y
   * ABS\_MT\_TOUCH\_MAJOR
   * ABS\_MT\_TOUCH\_MINOR

对 ABS\_MT\_TOUCH\_MINOR 的支持是可选的。不过，如果您的设备支持它，请确保分辨率设置正确。

1. 如需确认这些字段是否已正确设置，请运行以下命令：

$ adb shell getevent -li

1. 如需在运行时启用此功能，请运行以下命令：

$ adb shell device\_config put input\_native\_boot palm\_rejection\_enabled 1

1. 重启 system\_server 进程。

$ adb shell stop && adb shell start

1. 确认 adb shell dumpsys input 会显示 UnwantedInteractionBlocker 中有手掌拒绝器。如果没有，请检查与输入相关的日志，以查找有关哪些方面配置可能有误的线索。

请参考以下示例：

UnwantedInteractionBlocker:

mEnablePalmRejection: true

isPalmRejectionEnabled (flag value): true

mPalmRejectors:

deviceId = 3:

mDeviceInfo:

max\_x =

max\_y =

x\_res = 11.00

y\_res = 11.00

major\_radius\_res = 1.00

minor\_radius\_res = 1.00

minor\_radius\_supported = true

touch\_major\_res = 1

touch\_minor\_res = 1

mSlotState:

mSlotsByPointerId:

mPointerIdsBySlot:

mSuppressedPointerIds: {}

1. 如需永久启用此功能，请在 [init\*\*rc](https://android.googlesource.com/platform/system/core/+/master/init/README.md) 文件中添加相应的 sysprop 命令：

setprop persist.device\_config.input\_native\_boot.palm\_rejection\_enabled 1

## 延伸阅读

1. [Linux 多点触控协议](http://www.kernel.org/doc/Documentation/input/multi-touch-protocol.txt)
2. [Linux 上可用的多点触控设备的 ENAC 列表](http://lii-enac.fr/en/architecture/linux-input/multitouch-devices.html)