**P7**

2021-10-5 20:4:24

【问】

好，那我们现在就开始吧，你首先大概简单的介绍一下你的背景以及你做过的一些开发的经历，就是说一下你大概做了多少年的开发。

【答】

哦，我现在是研一，然后本科的话是就读于XX大学的XX学院，现在改名叫XX学院，然后XXXX专业，XXXX专业是软件工程大类下的，所以我们接触的开发其实比较多的，课程设计的话，比如说像软件，我们还有一个课叫软件开发，然后还有一些比赛，就比如说一些课，比如说什么像编程，C++，然后像数据结构，这些是编程语言上面的东西，然后自己开发的课程，比如游戏导论，开发游戏，比如说交互设计，开发交互设计的软件，然后比如说，刚才说的软件开发那个课是用android studio去开发一个APP。然后还参加一些比赛吧，就比如说XXX开发大赛，然后做过一些网页、前端，这都算开发吧，然后还有一些项目，跟老师做的，比如说用unity来开发一些PC端的软件，这样子。

【问】

大概有多少年的开发的经验？

【答】

是从本科开始到现在吧，就是4年这样子。大概4年。

【问】

嗯好，那你可以就具体的讲一下你学过的某一个软件或者是某一个工具，某一个开发的框架，你大概讲一下这个过程，你是怎么学的，从你听说这个工具或者框架开始，就决定学它开始，到学会了它，然后用它去做某一项开发，做出某一个功能来，这个具体的学习过程是怎么样的？

【答】

那就拿unity举例吧，因为用的比较多，然后unity的话因为首先是我们学校是使用这个软件的，所以呢，在上课的过程中老师就会有一定的教学，然后老师一定的教学，这属于开发文档吗？老师的上课教学演示视频，这一类的可能就不算开发文档。

【问】

视频的话算一种教程，也算吧。

【答】

那如果也算的话就是老师的上课的一些教程视频，然后课后的话，unity它有官网，然后它有自己的那个forum，然后都会在上面有一些问答，然后就是基本的操作学会之后，就是具体的根据你想要开发什么功能去找对应功能需要的一些，具体的一些对应的代码语句这样子，比如说我开发一个游戏，然后在开发这游戏的过程中，首先技能操作是从老师那里学到的，学到之后可能有一些是在基础操作之外老师没有讲过的内容，就是上网去查一下相关的一些，比如csdn，或者是一些forum，它会有一些问答的东西。

【问】

你刚才提到的这个软件它有一个官网是吧，那官网上是有一个类似于它的官方文档的这样的一个资料，是吗？

【答】

嗯对，但是我其实不怎么看它的官方文档的资料，完全是看论坛上的讨论，这样子，就是直接去问，比如说这边怎么做，然后呢，就会论坛上面可能有人有相关的问题什么的。。因为一般10分钟遇到问题了我们才会去看那个，如果自己能解决的话就自己直接开发了。

【问】

嗯你做的事情其实是相当于基于你老师的教学，然后就老师教完之后，你其实已经对这个东西有一定的掌握了，然后可以直接用它去做开发的，只是说在遇到问题的时候你才会去查是吧，嗯，那你可以举一个你自学的例子吗？不是由老师教学的，是你完完全全靠自己自学，然后学会它，用它去做开发的一个例子。

【答】

哦，不是有老师教学，我想想，可能前端开发这个相对来说老师没有太教学吧，主要还是靠自己去，因为当时是我想参加一个比赛，然后我们几个同学就是想一起开发一个网站，然后有同学负责后端，它们主要是让我负责前端这一部分，我就去学了一些，就是先去了解了一些比如说前端的一些框架像Vue，bootstrap这些，还有它涉及到的编程语言，先是做了一个整体大概的了解，需要什么东西，然后呢再根据具体的，我当时可能首先是需要一个，因为我们不一定要从头开始做，所以需要一个，可能已经大概有了一个模板，然后呢，所以先去找一些比较合适的这种模板，然后找到之后再根据我们的具体的需要继续加工。像比如说菜鸟教程当时看挺多的，就是教程，然后它上面有非常详细的介绍。然后你其实只要看了大概前几个之后就大概能掌握它的基本操作了，就没有太大问题，然后具体的一些比如说交互上的具体的应用，你当时要再用的时候，你再回头去具体看一下那个对应的教程就可以了，所以大概就这样就能够做出一个还算可以的东西。

【问】

那你一开始选这个框架，比如说你刚才提到的一些前端开发的框架，你在选这个框架的时候，应该也是参考了一些资料吧？

【答】

嗯，对，会了解一下现在最流行的一些框架，哪些是比较好用的。

【问】

是怎么了解的，是有介绍的文档吗？

【答】

嗯，这个我倒不是看文档，因为其实蛮多是靠，就是我觉得沟通很重要吧，就比如说会去知乎上搜或者是跟学长学姐了解这样子。

【问】

所以就决定选用这么一个框架，选定了之后，学习它也主要是看教程，教程就是像菜鸟教程这样的一种技术的网站是吧，比较简单的那种教程，然后看完之后就基本上能够掌握了就上手做，然后在遇到问题的时候再去查教程，对吧？

【答】

对，因为其实像HTML或者就是前端开发这种，我们有一定编程经验的人，看到它的那些内容是大部分能理解的，其实，就是你有一定的经验，你看一些东西不会觉得，虽然它是完全不同的语言，但是你大概它是怎么回事儿，所以其实没有那么，就不会说是完全零基础上手的那种感觉吧，嗯，就是所以大概了解一下。

【问】

就是前端的语言，像CSS也是一看就基本上能明白了。

【答】

对，它比较简单。

【问】

大概了解了，接下来就是问一下你，你觉得在你的开发的过程中，文档扮演什么样的角色，它是占据着比较重要的位置吗？

【答】

嗯，我的开发，如果遇到问题的话，一般就是两种方法，要么就是求助学长学姐或者求助身边懂得的人，要么就是求助于技术文档。首先应该是求助技术文档，通常是这样子的，然后嗯，所以其实是蛮重要的角色，如果没有网上那么多相关的关于各种问题的解答，我觉得很难快速的开发出一个完全不懂的一个东西，或者是刚了解完就要去做出来。

【问】

嗯，也就是说在你的开发过程中文档还是比较重要的，然后你遇到问题的时候是优先去求助文档或者是去网上搜索答案，然后再去问别人，对吧？那你是怎么就是怎么找到这些资料的，除了搜索，有没有其它途径获取你需要的答案？

【答】

当然了，就比如说就是看一些类似的人家已经开发出来的东西，对吧，你想用这个功能，人家开发出来了，你就借鉴这个功能。

【问】

人家已经开发出来的东西是是一些模板是吗？

【答】

对，我有点不太理解，就是除了搜索引擎，它到底包含了多大的一个范畴？

【问】

可能是你就知道那个东西，然后直接进到它的官网，比如说输入它那个链接就直接进入它的官网去找你想要的资料，这个就不算使用搜索引擎，有可能是像前面的人提到了它们会有什么，就是程序员的社区，比如说，会有某个QQ群，通过这种QQ群也可以获得这些资料。

【答】

主要还是搜索，然后你刚刚提到的通过一些链接进入，可能也会吧，就个别，但这个其实是个别的，相当于是你已知有这样子的一个，比如说我刚刚说的unity的论坛，就是可以直接通过论坛里面去查相关的内容。

【问】

了解了，那就是你在找到这个东西之后，相当于你打开了这个文档之后，你是怎么阅读的呢？就是大概一个阅读习惯是怎么样的？

【答】

我是带着问题，如果我带着问题来的话，我会直接找我要的答案，它不是一般都有目录吗？就先找到我要什么，就直接点进去看我要的那一部分的文档，这样子看下来，然后看能不能解决我的问题。

【问】

就是点开它那个导航或者它那个目录，点到对应的章节，然后去看。那快速定位，你会使用这种搜索的功能吗？内部的搜索或者是ctrl+F。

【答】

也会使用的。

【问】

那你可以回忆一下你都用过哪些类型的文档吗？

【答】

嗯，我就刚刚提到的，基本上都用过，比如说通过搜索引擎搜索，比如说百度谷歌这些，然后去看，一般搜出来的，要么就是一些别人写好的博客嘛，就比如说各种博客都有可能吧，要么就是一些官方的文档，要么就是一些别人提的问题，差不多就是这种情况，然后别人写的教程我觉得是可能看的比较多的吧，因为我搜索的时候搜索关键词，可能就会比如说，我现在需要做一个什么游戏，我就会搜，比如说unity，然后小球弹跳，比如说这样子。它出来就会有小球弹跳的各种教程。那有可能会搜出来Unity它自己的官网在介绍关于比如说弹跳制作，动画制作之类的。然后还有就是在csdn上的一些博客下面的回复，其实也挺那啥的，就是经常看完博客之后出现一些问题跟它讲的不一样，但底下可能评论的同学它们也有类似的疑问，然后就会在那里去寻找答案，嗯，然后要么就是，一些比如说需要下载的文档啊之类的内容，它不是直接搜出来的，它可能是一个链接给你，然后你去比如说网盘上下载啊或者什么的，然后是一个技术文档这样子。

【问】

这种是官方的文档吗？

【答】

不是，一般由程序员自己写的，但是可能会需要用钱啊之类的下载。

【问】

这种相当于它们自己写的笔记，是吧？

【答】

笔记，付费笔记。但是我一般有时候也会用，因为有时候找不到那种，然后还有就是向一些朋友求助啊，感觉在文档上，就是基本上就是看官方和教程这样子，还有一些非官方的教程。

【问】

嗯，那就是你有接触过国外的一些文档吗？国外的一些东西的文档，像刚才提到的Unity应该也是国外的。

【答】

对，它的文档就是全英文的。

【问】

也是接触过国内国外的一些开发的文档对吧，那你觉得它们的质量怎么样？你可以举一个比较具体的例子。就举一个比较好的例子和一个比较差的例子，然后说说它们这个文档好在哪里，不好在哪里。

【答】

国内我使用过的软件的官方文档比较少，我只使用过菜鸟这样子的非官方文档。

【问】

那你可以说一下菜鸟教程，你觉得它好不好？好在哪里，不好在哪里。

【答】

像菜鸟教程的话，我觉得它比较好的一点是可以实时的对你的学习进行一个，它会有一个可以让你直接运行使用的一个代码块，然后就可以直接知道，直接生成结果，然后可以自己改变一些代码，然后看它的结果输出。它其实是一个非常偏教学性的东西，它就是让你学会这个东西，但是我觉得其实很多国外的像这种软件的开发文档，它更多是你不懂然后去查，就像一本书一样，它不是要你去阅读完整篇文档，然后教会你一个什么东西，而是你有问题的时候来找它，然后它会帮你解决，我觉得这是一个挺大的区别。像菜鸟它就是每一个内容都写的很简单，其实一段话可能就刷到底下，没有几行。它会有具体的代码段，然后一些很简明扼要的方法告诉你，这个语法怎么用，类似这样子。但是如果具体的国外软件文档，它就会讲的更详细，然后它可能会配一些视频教程。但是菜鸟不会有视频教程，通常。就是说我觉得菜鸟它比较好，可能就是好在，一个就是我刚说的比较简明。如果你想去学一个东西的话，就很快上手，因为你可以在里面直接进行操作，不用自己再出来开一个新的程序，然后运行看结果，你在里面就可以操作，然后也很容易学嘛，很容易上手。然后缺点的话可能就是它其实没有那么完整，它其实很基础的跟你讲一些内容吧，例子也比较简单，然后你觉得就算你全部都看完一遍，你学的也不是很精，你要真的想要学精，反正我的想法是，如果你真的要想学精的话，你还得去看它的官方文档，其实我会有这种感觉，然后当你去官方文档的时候，它是会有非常详细的介绍，然后还会比如说配视频啊。但它不会有那种让你在现场直接尝试的功能了，一般就是你得自己在自己的软件上去试，但步骤什么的一般也会比较清晰，也会比较清楚。国外的这种软件文档，我觉得可能有一个缺点。可能是它会比较冗长吧，而且读起来可能没有那么易读吧，有的时候容易放弃，优点就是它比较完整，比较全面这样子。

【问】

嗯，就国外的，你可以举一个比较具体的例子吗？嗯，你比较喜欢的某一个文档。

【答】

喜欢的文档，我想想我上次用那个Android studio的时候，Android studio它有个官方网站嘛，我是为了unreal装的Android studio，那个unreal engine，一个软件。然后我去unreal的官方文档上面找它匹配装Android的SDK的相关的内容，然后它其实非常详细，就是它会把每一步截图出来，就会把怎么安装，你点这里，然后点这里，如果你出现什么情况，它会有一个warning，假设你出现什么情况，你可能要怎么怎么样，然后说这一步，如果你选了什么什么的话，你可能就。就是它非常非常的清楚，你几乎每一步都会找到对应的。因为其实配环境我们都知道特别痛苦，但它在配那个环境的时候，它就让我觉得很清楚，然后每一步就算是出现了问题，它大概有个方向说，那你可以去哪里去找问题，嗯，而且每一步都有截图，就不会像纯文字说明，有的时候你可能找不到或者怎么样，它每步都有截图，然后你就照着它那个摁就很清楚。

【问】

就是说它那个安装教程有比较有很清楚的截图，很直观，然后让你一看就能明白。对那除了它的安装之外，运用到它具体的功能那一部分的解释啊，有什么优点？

【答】

嗯，我觉得一个是清楚，然后一个就是它，嗯，我想想怎么说，优点的话，我感觉，就是你找到解答，就是你能在这里找到你想要的答案。

【问】

你遇到的问题都能在这里解决？

【答】

嗯，基本上能得到解决吧，嗯。

【问】

嗯，那除了内容本身，那这个文档它的结构怎么样？

【答】

结构好像也挺清楚的吧，因为我一般都是目的性挺强的，我都能找到我想要的东西，说明它应该还是，我其实没有仔细观察过它的结构，但是就比如说我要去找一个渲染的东西，我就直接能够大概，比如说有个菜单上面有个render，大概就知道这是渲染，然后我到下面去再看它的菜单，然后就这样找下去，基本上好像没有遇到太大阻碍过吧。

【问】

就是说它的结构还是分目录、子目录这样来编排的，所以比较清楚，可以很快的找到你需要的东西，是吗？那这个网站它会有像刚才你说的这个交互的东西吗？就有一行代码、一个例子，然后点一下能运行有这样一个这种交互的东西吗？

【答】

倒是没有，但有可能原因是，因为它是个软件的教程，所以可能不太像菜鸟的那么方便。

【问】

理解，哦，那你再举一个比较差的例子，就是你看起来这个文档体验很差的，基本上解决不了太多问题的文档的例子。

【答】

嗯，可能是别人写的文档吧。

【问】

可以具体说一下。

【答】

就比如说，我觉得我最近就经常遇到这种情况，因为我想要，比如说用unreal去做一个很简单的事情，然后我去搜索之后得到的一些教程，它们都写的非常的冗长，比如说我不知道像什么字号大小或者字体之类的，它就会写得让你根本没有看下去的欲望，然后它的文字又很多，内容又很杂，乍一看就感觉，它虽然是跟你想的标题是一致得，但是内容跟你想要的内容不一样，就会confused。

【问】

这个不是官方文档是吧，是别人写的？

【答】

对对，就是csdn上那种笔记，就是质量不太好。

【问】

嗯，那有遇到过质量比较好的吗？

【答】

也有。

【问】

像这种博客上、问答社区里面的东西有好的也有不好的文档，对吗？

【答】

嗯，当然。

【问】

那你还用过微信的开发者工具。

【答】

噢，对，但是那个也是参加一个比赛，用的不是特别多吧，大概也用过一点。

【问】

你觉得它那个文章怎么样？

【答】

微信的那个文档我印象，因为那其实我大二的时候，稍微有点久远，所以不是特别，我印象里它也是有一个侧边栏，然后侧边栏上面有挺多那种子目录什么的，就我现在的印象，我感觉还可以。

【问】

就还是能够用它这个开发者文档，然后能去做那个小程序得开发是吗？

【答】

嗯，我们当时好像用挺多开发者文档上面的介绍。

【问】

嗯嗯嗯，好明白了，那下一个问题就是，关于这个文档的反馈的的问题，就是说假如文档的下方它有一个反馈的途径，比如说会问你，您觉得这个对您有帮助吗？或者是你有什么意见吗？类似于这样一个反馈的途径，有可能是不同的形式啊，有可能是某一个按钮或者是要你提交一个表单，或者是进入到某一个提意见的这种页面，如果有这样一个反馈的途径，你会给它反馈吗？

【答】

我感觉我见过挺多这样反馈的途径，但是我从来没有过反馈，即使有的时候我真的很生气，我也不会反馈。

【问】

为什么呢？

【答】

好像就觉得唉呀，好烦了就关掉，就觉得唉呀没有用我就关掉了，好像并不会说因为没有用就去吐槽这件事情。

【问】

是觉得这个内容它本身对你没有用，所以你就？

【答】

对，我是指比如说看到别人的博客，好像有的时候底下是会有反馈，这个就是对你有用或对你没用嘛，我一般都也不太会去点。我有可能点的就是可能对我来说真的很有用，我可能会点个赞之类的。

【问】

有用的会点赞，没用的也懒得去反馈。

【答】

因为要写好多字啊。

【问】

那像官方文档你有见过这样的反馈途径吗？

【答】

这么说总感觉好像有，我觉得应该是有见过吧。官方文档，我可能没太反馈过，嗯，我想一想，应该是没有过，可能也没有让我特别不满的官方文档吧。

【问】

嗯嗯，了解了就是基本上没有做过反馈，那如果以后你遇到这种情况你会愿意给反馈吗？因为你的反馈有可能就帮助它文档背后的这个团队来提高它们的质量。

【答】

嗯，会吧。

【问】

了解了，嗯，那下一个问题是，你有写过技术文档，你自己有没有写过？

【答】

我写过软件的文档，就是开发一个软件之后就写文档，就包括软件的开发背景，然后开发使用的一些语言啊什么的，然后就是具体开发了什么，因为我们一般都是开发项目，工程项目，所以就是像工程文档那样子，就是会介绍说我们有哪些关卡或者是哪些页面，哪些功能这样子，然后每个功能对应了什么，当时怎么开发的，UI设计这些。

【问】

这个工程文档是给谁看的？

【答】

给使用这软件的人看的。

【问】

使用这个软件的人，那你们做这个软件是要面向市场的吗？

【答】

但我们并没有真正面向市场。

【问】

那你们内部的人会看这种文档吗？

【答】

我们内部的人是写这个文档。

【问】

那你们其实相当于只是完成一个模拟的项目，并没有真正的去投产使用，当然也模拟了这个写文档的过程，对吧？

【答】

嗯，对，软件专利就应该不算文档吗？不算是这种技术文档？

【问】

嗯。那你觉得写这个文档的过程中遇到了什么困难，难点在哪？还是说写下来就非常顺利？

【答】

感觉很顺利，因为基本上，如果是自己团队开发的嘛，因为基本上我刚才讲的那些内容你都自己心里清楚的很，用了什么哪些功能，功能代表什么含义，对吧，哪些框架这些都很清楚。

【问】

好，那我想想还有什么问题。假如有这么一个理想的状态，以后你的开发工作当中可以用到你非常满意的文档，那你觉得什么什么样的文档才是让你满意的文档？或者说什么样的文档才算一份好的文档，你期待用到什么样的文档呢？你可以大概的总结一下，你希望好的那几个点。

【答】

嗯，首先是能把，我们以软件为例哈，使用一个软件的文档，能把这个软件的基础操作能解释的很清楚的，就是它不能只讲一些或者只讲部分，反正就是尽量讲的全面又清楚，能让我比较容易的理解到这些基础的功能，因为它本来就不复杂，所以这些基础工作能够以比较清楚的简单的方法让我能够掌握。然后它最好的结构比较清楚，整个的搜索的逻辑比较清楚，能够让我快速的定位到我想要定位的一些比较难的问题，然后这可能是两个部分吧。然后像这些比较难的问题呢，我希望它也能有一些比较全面深入的展开，就是基本上问题能得到解决吧，这样子。然后最好还可以有一些可以讨论的空间，比如说底下会有一些评论啊，回复啊这些，因为有的时候文档没有提到的，或许能够在评论中找到答案吧，然后自己也可以提问或者是可以回答。

【问】

你是希望读者能够有一个评论反馈的地方？

【答】

因为我之前使用过挺多那种能在底下的评论找到挺多答案的，其实这样就挺好的，我感觉已经基本上能解决我想要的答案。

【问】

那你觉得像读英文的文档，你觉得困难吗？

【答】

嗯，其实还好，因为其它不会有什么太专业的名词。

【问】

就觉得语言上并不是一个太大的问题？

【答】

不是太大的问题，因为其实我们都真的读很多英文的文档，而且也读很多论文之类的，所以就还好。

【问】

那你觉得如果这种英文的文档，它有双语版，是必要的吗？

【答】

会更好，嗯，对，因为我觉得肯定有很多同学不愿意读英文吧，因为中文肯定更清楚，如果同时有中英，我肯定看中文啊，因为速度会更快，更清楚。

【问】

那这个是从内容上来说，那从结构上呢，你希望文档是什么样的？比较好的文档的结构应该是什么样子的？

【答】

我希望它有基础的使用和深入版的使用，这样大家可以比较能够定位到自己是初级者的身份还是一个长期使用的老手的身份，就能够更好的，因为其实很多东西，比如我想要去了解一个内容，但是我作为一个初学者，你跟我讲很多很深的东西，我也看不懂啊，所以你可以用一些简单的话跟我讲，OK我大概明白是怎么回事儿，然后呢我多了解一些，这个软件，我慢慢摸索，大概清楚之后我再去看深入的内容，就会更好用，就是最好能有菜鸟版和老手版这样子，然后呢在内部再进行一些结构的划分吧，比如说具体的软件有哪些功能，比如说从它的基础使用啊，然后它的各种程序编写啊，然后一些组件啊，然后像一些渲染啊，像一些场景啊这些。

【问】

就是说它的结构首先是要由浅入深，或者是，就是干脆划分成两个不同难度级别的版本，然后直接版本内部再进行一个划分，也是由概述，然后再到深入。

【答】

对，然后最好是能够清楚的那个关键词，能够让想要查到的人立马查到它。

【问】

那从文档的交互设计上来说，你觉得有什么比较好的呀？

【答】

交互设计的话我觉得就是左边就是交互栏，右边就是文字就挺好的，挺清楚，然后你可能滑动的时候它可以先缩进去，就先隐藏起来，然后你需要的时候它才弹出来，然后基本上就是像好多列嘛，然后你点开之后就在底下再列出来，然后你再点一下它就可以再回去，这样子，所以你可以很自如的掌握它的内容。

【问】

嗯，然后那个你刚才好像提到一个代码的运行。如果有这种，它的那个demo，它那个例子里面那个代码是可以直接就运行，不用你自己去手动安装环境的话这样也比较好是吧？

【答】

对肯定的。

【问】

嗯，还有其它的吗？排版上来说，你提到了一个最好有图片，图文并茂，是吧？

【答】

如果有视频更好

【问】

视频应该是更直观对，确实现在有一些文档，它们开始上手那部分会有一个比较简短的视频。

【答】

嗯嗯，对，就是一下就上手了，就不会觉得这个软件对使用者来说很难，你就会先有个底。

【问】

对你来说看视频比看文字更容易是吗？你更倾向于看视频是吗？

【答】

对，没错。

【问】

还有其它要补充的吗？

【答】

没有了。

【问】

问一个跟文档本身不太相关的问题，因为我们这个研究它需要面向研究的是中国的开发者，然后就是想了解一下中国的开发者程序员对于自己的一个定位吧，你觉得在中国，程序员这个群体它和其它专业其它职业的人相比，它有哪些不同的点，有哪些特征？

【答】

我觉得可能，压力更大一点，工作压力更大，我觉得或许是吧，因为可能你需要面对很多的业绩考核呀，或者是面对很多，就是一直在瞬息万变的这个时代，程序员是最需要赶紧的跟上，然后去获取最新的资讯，然后去改变，比如说是开发也好，什么的也好，要一直不断的更新，不断的更新，更新很快的。

【问】

你是说工作压力本身也大，然后技术在不断的更新迭代，然后你们也有一个学习的压力？

【答】

嗯，对啊，而且有开发的压力，有学习有开发的压力。

【问】

那你觉得在学习习惯上程序员它和其它专业的人有什么不一样的？

【答】

我感觉我们好像都偏向于，就是地基也没有那么扎实，但是已经在垒房子的感觉。大家其实通常就是想做一个东西就去做了，但是其实很多东西都还不是很清楚，就已经在实现一些看起来很高端的内容，比如说开发一个网站，实际上你对它的代码本身或者对它框架本身也不够那么了解，但因为现在实际上有太多教程来教你一步一步跟它做，可能就能做出来个像样的东西，或许也可以应付你的老板或者说是你的工作，但真的要问你说你真的很懂一些具体的内容原理可能未必，但可能其它专业不会这样吧，或许他们都比较扎实吧，我猜。

【问】

那就是虽然你的知识，这个知识结构并不是很扎实，但能够上手去做，那做的过程中肯定也是遇到问题，那遇到问题的时候你们是倾向于自己去解决，是吗？

【答】

嗯，我其实会比较倾向于自己解决了，然后解决不了的话再求助同学这样子，

【问】

那学习的话呢，学习也是倾向于去自学吗？应该你们学习的大部分的时间都是在自学的。

【答】

对，学习都是自学的，只能自学。

【问】

没有其它的问题了，我们今天的访谈就结束吧。