开源通识——开源法律法规

1. 知识产权的基础知识

1.1 版权、商标和专利

知识产权是指自然人、法人或其他组织对其智力创作成果依法享有的专有权利。

知识产权可大致分为两类：

- 一类是工业产权，包括专利、商标、禁止不正当竞争、商业秘密、地理标志等；
- 另一类是版权（即“著作权”），涉及文学、艺术和科学作品，诸如小说、诗歌、戏剧、电影、音乐、歌曲、美术、摄影、雕塑、产品设计图、建筑外观、计算机软件等。

	版权 **Copyright**	**商标** **Trademark**	**专利** **Patent**
符号	©	®	™
内容	作品（保护的是作者思想、观点和情感的表现形式）	区别不同生产经营和不同生产该商品的商品和服务标记	发明、实用新型和外观设计（保护的是发明创造，属于思想/观点内容范围）
条件	独创性	显著性	新颖性、独创性和实用性
产生方式	自动生产	注册申请	申请
审批机构	软件、外国人作品、合同、其他作品 中国版权保护中心 其他作品 各地版权局	国家知识产权局	国家知识产权局
期限	自然人有生之年+去世后 50 年	10 年，到期可以续展	发明：20 年

	版权 Copyright	商标 Trademark	专利 Patent
	法人、组织：50 年 可继承，不可续展		实用新型、外观设计专利：15 年
起始时间	创作完毕日	审核公告日	申请日
地域	国际版权条约成员国	登记国（地区）	登记国（地区）
保护力度	只要是独创作品，基石雷同，也各自拥有版权	不保护相同或近似商标	不保护同样的专利

1.2 合同与版权

作品创作者的所有权利在该作品完成时成为他们的所有权利，然而创作者很少自己利用这些权利。大规模生产和发行作品历来是资本密集型的，因此作品的创作者通常依赖他人制作其作品的实体副本并进行发行。版权的概念与 15 世纪印刷机的发展同步发展，它最初保护印刷商排他性地使用他们委托的作品的权利。创作者和出版商之间进行的最典型的交易是作品的许可以换取付款，称为版税。

软件发行一般不涉及向个人支付版税。因为商业软件通常是由大型团队 制作的，需要大量的资本支出，所以由此产生的工作是“受雇工作”。此类作品的版权属于雇主，就软件而言，雇主通常也是软件本身的出版商和分销商。

2. 软件许可证的基础知识

2.1 什么是软件许可证?

软件许可协议（也称为最终用户许可协议或 EULA）是软件程序的所有者或开发者与用户之间达成的具有法律约束力的协议，概述了他们如何使用和分发产品。在这份文件中，明确了各方的责任，防止和保护开发者免受版权法的侵害。该许可证旨在保护软件开发商的知识产权，并限制可能因使用该许可证而对他们提出的任何索赔。软件许可证还为软件的分发和使用提供具有法律约束力的定义。最终用户的权利，如安装、保证和责任，也经常在软件许可中详细说明，包括对开发者知识产权的保护。

开发人员出于多种原因发布软件，无论是为了展示新想法、为尽可能多的人提供利益，还是为了经济和经济利益。为了确保流程中涉及的所有各方都能从软件中受益，必须明确定义其使用条款和条件。

软件许可内容的详细基本规则：

- 可以安装软件的位置以及可以安装的实例数量。
- 如何使用该软件。
- 软件是否可以被复制、修改或重新分发。
- 适用于该软件的任何版权。
- 软件的所有权——通常指定提供商保留所有所有权。
- 协议条款的期限。
- 什么构成了软件的正确使用。

2.2 为什么软件许可很重要

在安装任何软件之前，它必须获得合法许可。否则，使用该软件将被视为违反版权法。对于充当用户的企业而言，了解软件许可尤为重要，因为您不想为未使用的许可证支付不必要的费用。虽然收购太多可能会浪费公司的资源，但收购太少

可能会使公司容易受到代价高昂的潜在诉讼。此外，软件许可协议可以使管理公司软件的方式更加轻松，因此找到合适的协议非常重要。

作为软件开发人员，软件许可协议保　　护时间、精力和资源的投资，确保您能够从辛勤工作中获利。获得软件许可协议的其他一些动机是因为它：

- 防止客户滥用您的软件：如果没有签署许可协议，就无法阻止您的客户为了自己的利益而尝试复制您的软件。通常，尝试复制软件的客户这样做是为了可以将其安装在多台计算机上，从而省钱，但他们也可以尝试出售软件以获取利润。无论哪种方式，这都可能会损失您的潜在收入。
- 允许您在不出售软件的情况下获得软件许可：您可以让客户为您的软件许可付费，同时保留对它的所有权利，允许您在限制其使用的情况下许可它。这实质上使您可以更好地控制软件的使用和分发方式，并让您有机会随着时间的推移赚更多的钱。
- 让您有机会拒绝保证：客户在购买产品时都有一定的期望，您的软件也不例外。通过在您的许可协议中包含免责声明，您可以要求用户接受可用的软件或按原样接受软件，从而让他们承担风险。
- 限制您的责任：限制您的责任非常重要，因为它有助于保护您的公司免受潜在的诉讼。
- 允许您自由撤销许可证：包括协议的这一部分，您可以随时出于任何原因撤销或暂停许可，帮助您保持对软件的完全控制。

2.3 软件许可证的类型

大多数软件及其协议属于两大类许可：

- 专有软件许可证：专有软件许可证限制用户更改软件代码的合法能力。很大一部分软件许可属于这一类，因为大多数商业软件许可限制最终用户的访问并定义可接受的使用条款。它们还为开发人员提供最大程度的保护。
- 自由和开源软件(FOSS)许可证：FOSS 许可证是允许最终用户共享、修改、使用和重用软件产品源代码的合同。FOSS 许可证授予用户对其使用的相当大的权限。

在这两种情况下，软件许可证通常会指定使用软件产品的责任限制、任何相互责任（例如支持）以及任何保证或保证免责声明。

开源和专有软件许可还可能指定额外的限制和条款：

- 所有权转让给买方或卖方保留所有权
- 任何复制、销售或分发软件的授权
- 许可证是否构成购买或租赁软件的定义

如果软件不在任何许可范围内，通常将其归类为：

- 公共领域软件——可免费使用，不受版权保护
- 私人未经许可的软件——例如仍受版权保护的商业应用程序

2.4 软件许可如何运作

软件的新用户通常会签订最终用户许可协议（EULA），该协议构成许可方（提供者）和被许可方（用户或企业）之间关系的法律定义。EULA 是一份合同，确立了购买者安装和使用软件的权利。每个 EULA 都包含一个条款，规定最终用户何时激活其条件。这可能是用户打开产品包装的那一刻，或者，例如，当用户单击同意接受 EULA 条款以访问它的按钮时。

2.4.1 专有软件类型

- 永久许可证是永久密钥，允许无限期访问——通常是指定用户。
- 订阅许可证意味着用户支付较少的费用，通常每月支付一次，以继续访问软件应用程序。
- 浮动许可证意味着数量有限的许可证按照先到先得的原则由更多用户共享。

还存在其他更高级的许可选项，例如基于消费的许可，其中对软件应用程序的访问受时间或其他一些可衡量的因素或操作的限制。

2.4.2 SaaS 软件类型

软件即服务（SaaS）等基于云的应用程序通常会在 EULA 中包含许可详细信息，包括：

- 个用户的月费或年费
- 协议期限
- 取消协议的条款
- 如果在协议期间取消，将收回任何费用

2.4.3 “白标”软件类型

软件许可的另一种用途是在软件开发商或公司授权以第二方品牌销售或分销软件的情况下。开发商保留所有权，但品牌重塑公司可以转售软件产品。这种许可方法称为“白标”。

3. 开源许可证的基础知识

3.1 什么是开源许可证?

开源许可证可以理解为特定软件或组件的创建者与用户之间具有约束力的法律合同，开源软件许可证管理其他人（除了原创者）如何使用、修改或分发软件代码。他们授予其他用户使用代码或将代码重新用于新应用程序或将代码包含在其他项目中的许可和权利。

开源代码的主要优点之一是它的可见性，这使得解决问题变得更加容易，并且在文档缺失或不正确时更好地理解某些东西是如何工作的。根据开源许可证的类型，您甚至可以修改原始源代码以根据您的需要进行调整或修复您发现的任何问题。许可证将决定这是否可能，以及在什么条件下。例如，您可能需要公开任何修改。

许多软件购买者——甚至是新开发者——误解了“开源”一词的意思是软件可以根据需要使用、复制、修改和分发。这种误解可能是由于混淆了开源与公共领域或共享软件，两者都可以免费使用和修改而无需特定许可或许可。事实是，在大多数情况下，开源软件都包含在几种类型的开源许可证中，并且不一定是免费的。如果没有任何开源许可，该软件组件即使公开发布在 GitHub 上也可能无法被其他人使用。开源许可证定义了用户可以和不能使用源代码做什么。

与供应商通常无法访问、复制或修改源代码的专有软件相比，开源代码允许在其他程序或应用程序中使用、重用、共享、修改和分发代码。但与专有软件许可一样，开源软件受各种法律条款和限制的约束，具体取决于有效的开源许可类型。因此，遵守开源软件许可条款非常重要。您还应该了解与使用开源软件相关的其他风险。

有一种误解：一个代码库，如果公开了，就可以自动归类为开源。这是一种误解。许多开发人员经常提出这样的问题：“如果我将我的 GitHub 存储库公开，它应该足以让它开源吗？”任何组织或个人开发的代码都默认享有合法版权。这意味着未经创作者许可，任何人不得使用、编辑或分发它。如果源代码在没有任何许可信息的情况下意外或故意公开，则不被视为开源。因此，在没有明确的开源许可的情况下使用任何公开可用的代码都不是一个好主意。此外，如果您愿意开源您的任何代码，请确保您使用合法的开源许可证对其进行跟进。这是让开发者使用、更改或分发它的最佳方式。

3.2 为什么我们需要开源许可证?

虽然开源与传统版权许可的操作不同，它允许公开分发和公开修改，但取消第二种限制可能更为重要。通过要求版权所有者为他们分发的程序提供用户可修改的源代码，并要求他们允许开发和分发衍生作品，开源许可使得对传统专有商业软件许可模型的三个实质性改进成为可能。

- 第一个，也许是最大的好处是创新：现在已经很好地证明，程序员愿意为开源项目做出 贡献，除了让程序更有用之外没有任何回报。开源作品。可以为给定工作做出贡献的程序员越多，该工作可能具有的价值就越大。
-

- 第二个好处是可靠性：许多程序员意味着可以调试给定程序的许多人。此外，好处不仅仅是数字之一。知识渊博的用户，亲眼目睹了特定应用程序的局限性或错误对程序运行的影响，通常比原始软件的创建者。与软件发布者相比，这样的用户几乎肯定有更大的动力来纠正给定 代码段中的此类缺陷，在软件发布者中，进行此类更正的建议不仅必须与其他可能更紧 迫的更正竞争，而且还必须与发布者自己的竞争财务或组织限制。
-
- 第三个好处是长寿：当商业许可软件“绝版”并且不再受到其发行商的支持时，通常无法更新软件或使其适应新用途。这样的软件走到了进化的死胡同。相比之下，开源许可软件可能会在一段时间内被废弃，但仍然会被后来发现其用途的用户恢复、改编或重写这种用途可能与最初的预期用途完全不同。
-

3.3 开源许可证类型

根据开放源代码倡议(OSI)，官方或批准的开源许可证是与开源定义保持一致的许可证。建议尽可能坚持使用 OSI 认可的许可证。开源许可证主要可以分为两类：Copyleft & Permissive，区别主要在于对其用户施加的限制和义务的程度不同。

3.3.1 限制性/COPYLEFT 许可

这些许可证要求派生作品使用与原始作品相同的许可证。版权作为一项法律，在未经原创者明确许可的情况下，对创造性原创作品的使用、修改和共享施加了限制。例如，考虑属于创作者知识产权的书籍、电影等。当作者根据 copyleft 许可发布任何创作时，他们就在声明其原创作品的版权。只要保持义务的互惠性，用户可以更改、共享等工作。如果一个人正在使用具有开源许可证的组件，那么个人或组织也应该将其代码开源。

Copyleft 附带不同范围的不同类型的许可证。一些 copyleft 许可证只允许您发布修改后的代码。而其他许可证可能需要您在相同的基本许可证下发布整个应用程序。

Copyleft 许可证的主要目的是确保项目的开源性质得到保留。它限制个人或组织获取源代码并将其用于其专有软件。

3.3.2 宽松型/PERMISSIVE 许可

Permissive 许可在本质上限制较少。他们可以自由使用、修改或分发源代码。这些许可证还允许专有的衍生作品。许可许可也被亲切地称为“一切皆有可能”。它们对其他人如何使用、修改或分发开源代码的限制极少。它们还具有最低限度的义务或回报期望。

3.3.3 开源许可证的差异化

我们知道有很多开源许可证。当添加到基本的开源属性中时，这些差异是许可证中阐明的特定条款的基础，赋予它们独特的性质。根据这些开源许可证的性质，允许或不允许用户使用、修改或重新分发源代码。

某些指定的条款或附加条件可以是：

- Copyleft 许可类型，具有弱或强属性
- Permissive 许可类型是弱或严格
- 在用户界面或源代码中添加版权声明的义务
- 授予许可证自动专利
- 考虑到被许可人是开源软件的用户以及软件的作者

Copyleft 和 Permissive 许可都允许用户自由地复制、修改和重新发布代码。然而，两者之间的主要区别在于对其分布的限制程度。

当一个人修改一个 Copyleft 软件时，他有义务将新修改的代码保持在开放源代码下，这样它就可以公开使用。然而，当一个人修改一个许可软件时，除了对原始工作给予信任之外，没有这样的限制。用户可以随心所欲地使用修改后的代码，也可以自由地使其成为专有软件。

来自 Permissive 软件的衍生作品可以作为 Copyleft 使用，但来自 Copyleft 的衍生作品不能作为 Permissive 软件使用。还应该注意的是，如果在软件中发现了任何错误，只有当它是 Copyleft 许可而不是许可许可时，创作者才会承担责任。

由于可以选择将开源软件转换为专有软件，Permissive 软件通常比 Copyleft 软件更受公司的青睐，他们可以利用这些软件来实现商业目的，而不必披露他们修改过的代码。

Copyleft 许可背后的主要思想是，如果用户正在使用任何免费内容，那么任何由它构成的内容也应该是免费的。然而，宽松的许可证并不把这种搭便车视为有害的东西，相反，他们在这个问题上的观点是通过拒绝限制搭便车者挪用软件的好处来寻求软件的最大效用。出于同样的原因，Copyleft 内容比许可软件更容易像病毒一样传播。由于许可许可下的衍生作品是在不同的许可下发布的，因此它既不持久，也没有传染效应。

3.3.4 合使用不同许可证集的代码的问题

我们知道有不同的许可证，有些是宽松的，允许用户将代码与不同的开源许可证基本代码结合起来。这可能会让开源许可软件与闭源软件同化。此类许可证的一个示例是 MIT 许可证。其他许可证可能更严格，源代码只能与许可证与前一个许可证类似的源代码组合。最终结果要求与原始许可证相似。通用公共许可证（GPL）是此类许可证的一个示例。

考虑这样一种情况，您希望将一个代码许可证与两个不同的开源许可证放在一起。有了根据您的需要使用软件的开源自由，您就可以做到这一点。但是，当您想要重新分发最终程序时，可能会有些困难。这个最终程序需要在两个不同的许可证下分发，这可能是个问题。

ZES 就是这种情况的一个例子。ZES 是 2005 年出现在 OpenSolaris 中的一种创新和先进的文件系统。这是根据 CDDL（通用开发和分发许可证）的条款和条件许可的开源软件，虽然它是开源的，但它不能被吸收到 Linux 内核。这是因为 Linux 的源代码是根据通用公共许可证条款分发的。由于底层许可冲突，无法重新分发使用 ZES 支持编译的 Linux 内核二进制文件。

这些冲突只有在开源项目的所有者或作者同意更改开源许可证或在许可证中添加某些例外条件时才能解决。

3.3.5 开源许可和专利

许可是否在其许可授予中明确提及专利？通过作为许可的一部分明确授予专利权，许可明确表示不排除其他人使用他们贡献的代码实践的贡献者的任何发明。一些许可被认为具有隐含的专利授权，即使文本中没有明确授予。

软件可以实现专利发明。例如，某些软件发明已获得专利，如果您想使用它们，您可能需要向获得专利的公司支付费用。作为保护，一些许可证明确规定，那些在某些特定许可证下使用代码的人有权使用贡献者在代码中包含的方法的任何专利，无需额外付费。

明确排除专利的许可	隐性专利许可和/或未明确提及的专利	明确包括专利的许可
– CC0–1.0 – 其他知识共享许可，例如 CC–BY–4.0 – BSD–3–Clause–Clear	– BSD–2–Clause 和 BSD–3–Clause – MIT – ISC – Apache–1.0 和 Apache–1.1 – GPL–2.0 和 LGPL–2.1	– Apache–2.0 – GPL–3.0 和 LGPL–3.0 – MPL–1.0、MPL–1.1 和 MPL–2.0 – EPL–1.0 和 EPL–2.0 – CDDL–1.0 和 CDDL–1.1

注意：有时，项目会使用未明确引用专利的许可证，但会在单独的文件中添加专利授权。 可以在 GitHub 上的 Golang 存储库中的许可证文件 (BSD–3–Clause) 和附加专利文件中看到一个示例。

4 COPYLEFT：开源软件许可的兴起

开发人员看待软件概念的方式直到 1958 年才开始。在美国数学月刊发表的一篇论文中，John Tukey 是第一个将例程、编译器和自动编程称为软件的人。他试图说明由电子管、晶体管、电线和磁带组成的硬件与用于操作系统的代码等“看不见”的元素一样依赖。 几年后，世界上大多数大公司都迎头赶上，公司开始对软件进行版权保护，以防止其被免费分发或修改。1961 年 11 月 30 日，美国版权局收到了北美航空公司提交的第一份软件版权申请。 这种趋势一直持续到一天；一台新打印机到达麻省理工学院，源代码突然成为专有的。它促使 Richard Stallman 推动保持源代码开放，最终在 1985 年发布了他的 GNU 宣言。

4.1 COPYLEFT 是什么?

> Copyleft 是一种使知识产权可以不受任何限制地重复使用和修改的方法，除了使用原始资产生产的任何新产品也必须免费提供。这可以适用于从艺术作品到软件的一切事物。

Copyleft 是一个术语，最早出现在 Li-Chen Wang 博士于 1976 年编写的 Tiny BASIC 程序中。被广泛认为是第一个免费软件，该程序的分发包括 Copyleft 和所有错误保留作为许可文本的一部分，这是一个明显的反对声明版权。Stallman 注意到并使用 Copyleft 的想法在 1988 年制定了第一个 GNU 通用公共许可证 (GPL)。

GPL 基于四项自由:

- 自由 0：为任何目的使用源代码的自由，
- 自由 1：进行修改的自由，
- 自由 2：与任何人共享源代码的自由，还有
- 自由 3：分享的自由在变化。

但是，应该注意的是，GPL 并不限制用户出售基于原始源代码的衍生作品;它只要求源代码免费提供给任何想要它的人。

根据 Copyleft 的限制性强度，分为两种 Copyleft 许可类型：

- 弱 Copyleft:这是一种自由软件许可证，它强制规定，在该许可证下授权的软件的源代码将保持在相同的弱 Copyleft 许可证下。但是，应该注意的是，仍然可以从不同许可证(也可以是专有许可证)下的代码链接到弱 copyleft 代码，或者以其他方式将其合并到更大的软件中。简单来说，弱 copyleft 许可只适用于原始的 copyleft 作品。
-
- 强 Copyleft:这些许可证比弱许可证更进一步，强制要求任何链接或以其他方式合并其代码的软件都必须在兼容许可证下获得许可，这是各种可用的开源许可证的子集。正因为如此，这些授权被称为“病毒式传播”。简单地说，强 copyleft 许可适用于它的所有派生作品和包中的软件组件。
-

该运动继续进行，并导致创建了当今主要的开源库和存储库。2016 年，Unicode 接受了一项提议，将 Copyleft 符号包含在 Unicode 11.0 及更高版本中。

4.2 Copyleft 与版权

版权法规定了作品所有者控制所有复制、修改和使用受版权保护的材料的权利。Copyleft 与版权相反，广泛用于开源开发，Copyleft 许可证允许创建者、贡献者和维护者对开源项目进行更改，并与社区共享。Copyleft 使用此法律执行使用自由(包括所有其他权利)，并授予后续用户修改、分发和使用软件的权利。虽然版权限制了自由，但 Copyleft 在所有基于原创作品的后续材料上强制执行使用自由。

尽管 Copyleft 许可证允许出售对开源代码的修改，但它取消了对接收者如何使用代码的任何限制，并要求开发人员与接收者共享所有源代码。Copyleft 明确授予后续用户相同的权利，如果团队发布软件，则根据许可证对源库保密是非法的。

4.3 主流的 Copyleft 许可证

Copyleft 许可证允许任何人修改和重新分发软件，只要在衍生作品中也保留相同的权利。

最流行的 Copyleft 开源许可证，按限制程度排序，是 AGPL、GPL、LGPL、EPL 和 Mozilla：

- GNU 通用公共许可证（GPL）保留许可证通知和版权条款，适用于商业、专利和私人使用。任何使用 GPL 代码的软件都必须在同一许可下分发其所有源代码。因此，如果您在软件中使用 GPL 代码（例如，通过使用 GPL 库）

并分发您的应用程序，则您的所有源代码都必须在相同的 GPL 许可下可用。此限制使 GPL 成为强大的 Copyleft 许可证。

- Affero GPL（AGPL）只增加了一个条款，但对某些软件来说是一个重要的条款。由于 GPL 许可仅在软件分发时触发，因此对于仅通过网络可用的软件存在漏洞，即未明确“分发”。AGPL 许可证通过包含一个远程网络交互条款来弥补这个漏洞，该条款触发网络上使用的任何软件的 GPL 许可证。
- 较宽松的通用公共许可证（LGPL）提供与 AGPL 和 GPL Copyleft 开源许可证相同级别的条款，包括保留版权和许可证通知。主要的变化是较小的项目或通过较大的许可作品访问的对象不需要分发较大的项目。此外，修改后的源代码不必按照适用于较大代码项目的相同条款进行分发。
- Eclipse 公共许可证（EPL）通常用于商业软件。使用 EPL，使用 EPL、非 EPL 甚至专有代码开发的软件可以组合和再许可——前提是任何非 EPL 元素作为单独的模块独立存在或对象。可以在 EPL 许可下进行修改，但必须在相同条款下发布。
- Mozilla 公共许可证（MPL）是限制最少的 Copyleft 开源软件许可证。只要根据 MPL 许可的任何代码保存在单独的文件中，并且这些文件与软件一起分发，他们就可以轻松地在闭源和/或专有软件中修改和使用他们的代码。MPL 还包括专利授权并强制保留版权声明。

5. PERMISSIVE：更加商业友好的开源许可

通常，可以修改、复制、添加、删除等 Permissive 许可下的软件，无需共享这些更新。开发人员可以获取许可软件，通过更改或添加使其成为自己的软件，将新版本保留给自己，或者如果他们愿意，也可以共享。如果您希望创建可以出售并向竞争对手保密的专有软件，那么这是一项主要功能——这也是许可许可流行的主要原因之一。

这些与使用许可代码相关的自由与 Copyleft 许可的更严格要求形成对比，后者要求许可软件的任何衍生作品只能在相同的 Copyleft 许可下分发。这意味着任何对原始软件进行修改或添加的开发人员在分发包含 Copyleft 许可代码的软件时，通常都需要共享这些更改的源代码。

5.1 PERMISSIVE 许可证的历史

第一个许可许可证通常被认为是 Prior BSD 许可证，它是第一个“正式”BSD 许可证（今天称为 4 条款 BSD 许可证）的先驱。这个“原始 bsd”许可证出现在 20 世纪 80 年代后期。大约十年前，加州大学伯克利分校的计算机科学家开始研究一种基于 unix 的操作系统，后来被称为 BSD(伯克利软件分发)操作系统。当其他机构开始使用这个操作系统时，一些开发人员开始修改和添加源代码，然后要求将这些更改合并到下一个版本中。

1989 年，Berkeley 的 Network Release 1 包含了最早的 BSD 许可的前身，基本上说明了代码可以重用或修改。毕竟，它已经是了。一年后，4 条款的 BSD 许可证正式发布。从那时起，3-clause、2-clause 和 0-clause 许可证加入了 BSD 家族。如今，三条款版本是最受欢迎的。

MIT 许可证是目前最常用的 OSS 许可证，它与第一个 BSD 许可证几乎同时开发。然而，它的起源故事有点模糊。在 Twitter 上回答一个问题时，参与创建原始许可证的两名计算机科学家吉姆·盖蒂斯(Jim Gettys)和基思·帕卡德(Keith Packard)讲述了他们对这次经历的记忆。两人都在麻省理工学院帮助开发了 X Windows 系统。

根据 Packard 的说法，他们在 1985 年发布 X 版本 6 时添加了一个许可证。事实证明，在这个许可下分发软件是一个巨大的挑战，这促使 Gettys 打趣说，他们应该简单地放弃它。然而，使其成为公共领域是行不通的，因为这样 IBM 就不会使用它。因此，他们咨询了麻省理工学院的律师，起草了一份声明，表明任何人都可以使用和修改该软件。

随后的版本包含了类似的措辞，该许可证被称为 X 财团许可证。事实上，当开放源码倡议(OSI)在 1999 年分享其第一批已批准的 OSS 许可证时，它注意到 MIT 许可证有时被称为 X 联盟许可证。然而，X 联盟许可证和 MIT 许可证的“官方”版本(于 1998 年发布)并不是同一个许可证，它们使用不同的语言来传达类似的情绪。

尽管如此，现在的 MIT 许可证的前身似乎在当前版本编写之前已经存在了近十年。如果是真的，这将使它成为(如果不是)最早的开放 OSS 许可证之一。

5.2 宽松和限制许可的主要差异

Permissive（宽松）许可和 Copyleft（限制）许可两者之间的主要区别涉及许可证合规性条件以及对代码的任何更改必须“公开”的程度。

一般来说，Permissive 许可只要求用户在许可软件的任何再分发中包括两件事：原始版权声明和许可文本的副本。Copyleft 许可证也有这个条件。但是，它们还要求用户向二进制文件的所有接收者提供任何修改的源代码——并将这些修改置于相同的许可之下。因此，代码的所有修改最终都与原始代码完全一样“开放”。

需要注意的一件事：虽然遵守许可许可的条款通常相对容易，但许可下的软件与公共领域的软件之间存在差异。虽然有等效于公共领域的开源软件许可证（例如 BSD 0–Cluase 许可证，或知识共享 Creative Commons），但许可许可证不属于此类。一般来说，公有领域的作品没有使用要求，而宽松许可则有规定，无论多么微小。

5.3 PERMISSIVE 许可用例

为特定软件项目选择正确的许可证取决于许多因素，例如您希望其他开发人员如何与您的代码交互，以及您对将代码更改设为专有/封闭源代码的开放程度（双关语意）。

尽管如此，在某些一般情况下，许可许可可能是最合适的选择。例如，如果您满足以下条件，许可许可可能最适合您的项目：

- 想让其他人（包括大型商业企业）使用您的代码变得简单和有吸引力。毕竟，如果一家知名科技公司不能保持自己版本的专有性，他们可能不想修改您的软件。
- 需要您的项目与 GPL 许可代码兼容。例如，BSD 或 MIT 许可证与 GNU GPL 许可证系列的每个成员都兼容。
- 在倾向于使用特定许可的社区中创建项目。在这种情况下，同伴压力是一件好事。 例如，如果其他人都在使用 MIT 许可证，您可能也应该使用。
- 不想花费时间或资源来管理复杂的许可证合规性问题。有了许可，开源用户无需担心源代码公开要求。

5.4 PERMISSIVE 许可类型

与分为“强”和“弱”两种类型的 Copyleft 许可证不同，Permissive 许可证并不能完全归类。最流行的宽松开源许可证是：Apache、MIT 和 BSD。

- Apache 许可证需要分布式代码的许可证通知和版权和/或软件中的通知。但是，衍生作品、较大的项目或修改在分发时允许带有不同的许可条款，并且不需要提供源代码。Apache 许可证包含专利授权。
- MIT License 以其起源地著名大学的名字命名，可能是世界上使用最广泛的开源许可证，也许是因为它非常简短、清晰且易于理解。它允许任何人使用原始代码为所欲为，只要原始版权和许可声明包含在分发的源代码或软件中。它消除了作者的任何责任，并且没有明确包含专利授权。
- Berkeley Source Distribution (BSD) License 是另一种宽松的开源许可证，它保留许可证声明和版权，但允许在没有源代码和不同许可条款的情况下分发更大的或许可的作品。BSD 2-clause 许可证与 MIT 开源许可证非常相似，而 BSD 3-clause 和 BSD 4-clause 许可证增加了更多与重用和其他条款相关的要求或限制。

许可许可证在开源社区中越来越受欢迎。超过一半的开源项目使用 MIT License 或 Apache License 2.0，大约三分之二使用某种类型的许可。许多备受瞩目的 OSS 项目正在使用宽松许可证，增加了这些许可证受到的关注。例如，MIT 许

可证被 Ruby on Rails、jQuery 和 Angular.js 使用，而 Apache 2.0 是 Kubernetes、TensorFlow 和 Swift 的首选许可证。

6. 了解强开源许可证

一般来说，早期的开源创业公司会选择非常宽松的许可证，比如 MIT 或 Apache，但在后期会转向更严格的许可证，比如 GPL。但是时代在变，开源项目变得越来越“精打细算”。创始人从第一天开始就将开源项目作为业务来开展。最终，这会影响他们对授权的态度。我相信在不久的将来，我们将看到许多开源公司转向重型许可证，甚至从一开始就采用它们。

开创了这种强授权趋势的标志性开源公司:ElasticSearch、MongoDB 和 MariaDB。他们为什么决定这么做?他们的许可证有什么不同?它如何影响他们的社区?

6.1 MongoDB 许可(SSPL)

MongoDB 是第一个从许可许可证转换到 copyleft 许可证的主要参与者。2018 年 10 月，MongoDB 宣布过渡到名为 SSPL 或服务器端公共许可证的新许可证。你可以在这里找到关于它的原始博客文章。

SSPL 许可证是一个非常强大的 Copyleft 许可证，它基本上是 GPL 的重版本。SSPL 要求任何将软件功能或其修改版本作为服务提供给第三方的人，必须在 SSPL 下发布所有源代码，包括软件、API 和用户自己运行服务实例所需的其他软件。换句话说，这意味着如果您发布托管在 AWS(或任何其他提供商)上的

SSPL 许可软件的 SaaS 版本，那么您将不得不发布托管提供商的所有源代码。因此，这意味着您不能使用 SSPL 许可的代码创建与之竞争的 SaaS 解决方案。除非你有自己的开源主机提供商。

您可以看到，SSPL 许可证的主要目的是保护公司免受竞争的 SaaS 解决方案的影响。一段来自官方常见问题解答的摘录证实：SSPL 第 13 节的 Copyleft 条件仅在您将 MongoDB 的功能或 MongoDB 的修改版本作为服务提供给第三方时适用。其他使用 MongoDB 作为数据库的 SaaS 应用程序没有 Copyleft 条件。

何时以及如何使用 SSPL？SSPL 被 MongoDB 和 Elastic 等公司使用(该公司有两种许可证可供选择)。SSPL 非常有名，它提供了非常准确的语言来保护您的软件不被转售。主要缺点是文本太长，可能会吓跑贡献者。

6.2 Elastic 许可证（EL）

2021年1月，ElasticSearch 宣布从 Apache License 2。0 切换到 Elastic License。这一决定主要是由于 AWS 多年来一直在提供完全基于 Elasticsearch 的 Amazon Elasticsearch 服务。

如果您熟悉 Apache License 2.0，您就会知道 AWS 有权通过托管服务销售 Elastic。但是，他们仍然没有权利使用 Elastic 的商标，因为它在 Apache License 2.0 中有明确的说明。AWS 违反了这一条款，并造成了混淆，即 Amazon Elasticsearch Service 是 AWS 和 Elastic 之间的合作伙伴关系。

这些事件的组合导致 Elastic 决定更改许可证。实际上，该公司同时采用了 SSPL 许可和 EL 许可–用户可以自行选择选择哪个许可证。但这并没有阻止 AWS。他

们只是在 Apache 许可下分叉了 Elastic 的最新可用版本，并将其重命名为“Amazon OpenSearch Service”。

1）ELv1

Elastic 许可证（ELv1）是弹性 License 的初始版本。它最初于 2018 年推出，用于 X-pack 目录下的 Elastic 源代码。在 2o21 年 1 月，这个许可证被应用到所有的 Elastic 代码库中，但是它很快被更改为 Elastic license 2.o。如果你感兴趣，你可以在这里找到 ELv1 的文本。基本上，这些 ELv1 和 ELv2 几乎是相同的——ELv2 只是更短一点，更宽容一点。

2）Elv2

Elastic 许可证（ELv2）是一个非常简短和简单的非 copyleft 许可，允许使用、复制和分发软件。它只有三个限制:

- 您不能将软件作为托管服务提供给其他人
- 您不能侵入许可密钥功能或删除/模糊受许可密钥保护的特性
- 您不能删除或模糊任何许可、版权或其他通知

ELv2 和 SSPL 的局限性实际上非常相似，除了一点:SSPL 有 Copyleft 条款，ELv2 没有。理论上讲，使用 SSPL，如果您开放所有基础架构代码(Copyleft 概念)，您就可以将软件作为托管服务出售。ELv2 没有 Copyleft 条款——它只是禁止在任何情况下将软件作为托管服务出售。

何时以及如何使用 ELv2？ELv2 显然被 Elastic 所使用，并且已经被 AriByte 和 OpenReplay 等初创公司所采用。ELv2 已经被“年轻的”项目所采用，这是一个有趣的迹象。ELv2 对于“重型”基础设施产品是一个很好的选择，这些产品有被

托管公司转售的风险。ELv2 保护他们不受这个问题的影响，同时它不禁止将该软件作为不同产品的一部分使用。

6.3 MariaDB Business Source 许可证(BSL)

MariaDB 公司在 2017 年推出了名为 BSL 或业务源许可证的新许可证。此 license 用于 MariaDB MaxScale 智能数据库代理。MariaDB 服务器是 MySQL 服务器的一个社区分支，使用 GPL 许可。您可能还记得，GPL 是一个 Copyleft 许可，所有基于 GPL 许可的软件的衍生作品也必须按照 GPL 进行许可。这就是为什么 BSL 只应用于 MariaDB MaxScale，而不是 MariaDB 服务器本身。

BSL 许可证是一个棘手的问题。默认情况下，它不允许该软件用于商业用途，但保证在未来该软件将成为开源软件(Apache 许可的 GPL)。

- 无生产用途

基本上，BSL 禁止软件的生产使用。虽然有限的生产使用可以通过“额外使用补助金”。无论您是想将 BSL 软件作为服务提供，还是只是将其作为不同服务的一部分使用，这都无关紧要。BSL 默认情况下禁止这两种用例，除非“额外使用授权”中允许某些内容。

- 额外用户使用授权

这是一个非常灵活的结构，允许授予用户额外的自由。例如，看看 MariaDB MaxScale 许可证中的“额外使用授权”条款:

```
Licensor:             MariaDB Corporation Ab

Licensed Work:        MariaDB MaxScale (TM) v.6.1.

                      The Licensed Work is (c) 2021 MariaDB Corporation Ab
```

```
Additional Use Grant: You may use the Licensed Work when your application
                      uses the Licensed Work with a total of less than three
                      server instances for any purpose.
```

因此，如果您的服务器实例少于三个，则可以自由地将 MariaDB MaxScale 用于商业目的。

- 更改日期

关于 BSL 的最后一个重要部分是它保证在未来的某个地方变成开源许可证。让我们再来看看 MariaDB MaxScale 的许可证:

```
Change Date:          2026-03-08

Change License:       Version 2 or later of the GNU General Public License as
                      published by the Free Software Foundation.
```

“更改日期”条款有效地说明了软件更改许可证的时间。如果没有指定日期，则从代码发布之日算起，它等于 4 年。

变更许可证条款指定了将在变更日期生效的许可证。变更许可证可以是任何与 BSL 竞争的许可证，如 GPL 或 Apache 2.0。

有趣的事实是，更改日期可以更改很多次，只要你想。这实际上允许为不同版本的软件设置不同的日期。例如，MariaDB MaxScale 2.0 已经是开源和 GPL 的，但 MariaDB MaxScale 6.2 计划在 2026 年才成为开源。

何时以及如何使用 BSL?像 MariaDB, Sentry 和 CockroachDB 这样的公司正在使用 BSL。也许，它是现有的强许可证中最灵活的。对于功能从一个版本到另

一个版本发生重大变化的公司来说，这可能是一个很好的选择——新版本将受到商业用途的保护，而旧版本将在许可许可下自动发布。

6.4 Apache 2.0 的 Commons Clause

在强许可列表中，最后但并非最不重要的是带有 Commons Clause 的 Apache 2.0。它就是著名的 Apache 2.0 许可带有一个很短的序言，称为“公共条款”。让我们来看看 n8n 回购中这一条款的语言:

> “Commons Clause” License Condition v1.0
>
> The Software is provided to you by the Licensor under the License, as defined below, subject to the following condition.
>
> Without limiting other conditions in the License, the grant of rights under the License will not include, and the License does not grant to you, the right to Sell the Software.
>
> For purposes of the foregoing, “Sell” means practicing any or all of the rights granted to you under the License to provide to third parties, for a fee or other consideration (including without limitation fees for hosting or consulting/ support services related to the Software), a product or service whose value derives, entirely or substantially, from the functionality of the Software. Any license notice or attribution required by the License must also include this Commons Clause License Condition notice.

基本上，这个条款说的是你可以用软件做任何事情，就像下面在 Apache License 2 中说的那样。但您不能“出售”此软件。它明确禁止为本软件提供付费托管、咨询或支持服务。

公共条款的好处是，人们可以出售“实质上”不同的软件衍生版本。我稍后会澄清这一点。添加了公共条款的 Apache 2.0 不再被 OSI 视为开源许可证。简而言之，它是许可和 copyleft 许可之间的妥协，但它有一些天生的缺点。

- Commons Cluase 公共条款

Commons 条款最初由知名开源许可专家希瑟·米克尔(Heather Meeker)起草。理论上，该条款可以应用于不同的许可许可证，如 MIT、Apache 和 BSD。但 Apache 仍然是最常见的选项。

公共条款条款创造了一个时髦的术语“实质性衍生品”。也许，它应该被解释为对销售衍生产品的限制，这些衍生产品只会给软件增加一些无形的价值，比如更改名称、api 或函数签名。正如您可以想象的那样，这个定义可以被非常不同地对待，因此在实践中，在付费产品中使用 Commons Clause 许可的软件是相当危险的。

公共条款的另一个问题是混乱。当开发人员看到 Apache 2.0 时，他们等待 Apache 2.0。但是，公有条款把一切都颠倒过来了，所以 Apache 2 基本上什么都没有留下。尽管它的文本紧随下议院条款之后。

何时以及如何使用带有 Commons 条款的 Apache 2.0 ?带有 Commons Clause 的 Apache 2.0 被 n8n 和 Redis Labs 等公司使用(用于某些模块)。显然，Commons 条款可能是一个“软”选项, 可以从一个完全允许的许可证过渡到一个更强的版本。

6.5 对社区的影响

这里提到的所有许可证的一个共同缺陷是社区影响。它们不被开放源代码倡议(OSI)认可为开源许可证，因为它们违反了开源软件定义的第 6 条。这一条款基本上是说，开源许可证不得限制任何人在特定领域或努力中使用该软件。

三个重要方面可尽量减少对社区的影响:

- 许可证应该简短、简单、清晰。
- 它只应该禁止直接转售该软件。不应禁止将该软件作为不同服务的一部分使用。
- 它不应该使开发人员感到困惑。

7. 知识共享许可

知识共享(Creative Commons)许可证是允许自由发布其他版权“作品”的公共版权许可之一。当作者希望给其他人分享、使用和构建其作品的权利时，可以使用 CC 许可。CC 为作者提供了灵活性（例如，他们可能会选择仅允许对给定作品进行非商业用途）并保护使用或重新分发作者作品的人免受侵犯版权的担忧，只要他们遵守规定的条件在作者分发作品的许可中指定。

有几种类型的知识共享许可，每个许可证因若干组合而不同，这些组合限制了分发条款：

- BY：要求作者陈述
- SA：任何衍生产品都必须“相似地共享”，所以在相同的许可条款下——这就是病毒式传播模块
- NC：禁止商业使用——如果包含此模块，则许可证在 OSI 定义下不再是开源的
- ND：禁止衍生品——如果包含这个模块，许可证在 OSI 定义下也不再是开源的

8 公共领域，CC0，未经许可

您可能还会遇到已经发布到公共领域的软件，这意味着所有的版权主张都已丧失。这个概念似乎对那些只想释放代码、根本不关心代码会发生什么的人很有吸引力。

值得注意的是，尽管乍一看这似乎是一个相当迷人的概念，但它有一个巨大的缺点:没有保修豁免。你肯定不希望自己仅仅因为编写了一些代码而导致某些人出

现问题而受到法律威胁，所以如果你想走这条路，我的建议是至少考虑像 Unlicense 这样的东西，因为它至少包含了保证豁免，从而保护你免受责任。

9. 多许可

9.1 Dual License（双许可）

双重许可通常是指在专有许可和开源许可（通常是 GPL）下许可软件。虽然这似乎是一种相互矛盾的方法，但它已成为一种流行的方式，许可人通过这种方式获得与商业许可相关的经济利益，同时利用与开源许可相关的社区利益。对于被许可方来说，同一段代码既可以在专有许可证下发布，也可以在开源许可证下发布(同样，通常是 GPL)，这一事实可能会成为被许可方的陷阱，因为他们在遵从性和监督方面很容易会造成忽视。被许可方，特别是那些希望对一段代码进行商业重用的人，必须特别注意他们在适当的许可下使用代码。

使用双重许可，许可人可以在专有模式和开源模式下向被许可人分发软件，从而允许许可人同时利用两种类型许可的优势。也就是说，一些公司使用双重许可模式以两种不同的许可形式分发相同的软件：

- (1) 受专有许可约束的版本（可能附带进一步开发和商业分发该软件的权利以及许可方技术支持和附加功能），
- （2）根据开源许可证（例如 GPL）的限制和义务获得许可并受其约束的版本。

双重许可的一个著名示例是 Oracle 的 MySQL 数据库管理系统。Oracle 对 MySQL 使用双重许可模式来满足其消费者的需求。Oracle 在专有（OEM 样式）许可下为想要创建和商业分发包含 MySQL 的专有衍生作品而不泄露底层源代码并且不希望受 GPL 的其他限制和义务约束的被许可人提供 MySQL。此外，

Oracle 在 GPL 下向那些只想使用该软件或想将 MySQL 合并到产品中的被许可人授予 MySQL 许可，以便以后同样在 GPL 下分发。

对于受双重许可模式约束的软件（如 MySQL），有两种许可选项可供选择，被许可人必须：

- （1）在考虑和确定哪个许可最适合该代码的预期用例时确保提出正确的问题，
- （2）确保被许可人最终使用的代码受制于准确满足被许可人预期用途的许可。

不采取这些基本预防措施可能会产生重大后果。

9.2 多重许可

有些开源软件获得了两个甚至三个的授权。这意味着收到此多重许可软件的人可以选择在哪个许可下发布该软件。由于每个许可证授予不同的权限并施加不同的义务，因此必须做出选择，以选择最能满足需求的许可证。

这是一些库的常见情况。例如，NSS 是 Mozilla 开发的一个库，在其他与安全相关的特性中实现了 SSL/TLS 支持。它在 MPL、GPL 和 LGPL 许可证下进行了三重授权

9. 开源许可证遵从风险

以下是与开源许可证相关的一些主要风险:

- 误解许可义务：一个常见的错误是认为开源软件没有限制。然而，虽然软件许可证是免费的，但您必须承认并遵守它的要求。
- 法律后果：不遵守许可条款就是违反您的法律义务。竞争对手可以利用不合规来损害您的业务声誉，减缓项目进度，或使您损失资金。
- 安全风险：在不知情的情况下使用易受攻击或依赖的软件也会带来风险。您必须了解支持项目的技术，以避免引入缺陷。

许可证引入了不同程度的使用许可软件需要满足的条件。开源许可证确实提供了很多自由，但很少是完全无条件的：

- 几乎所有许可都要求您为分发的软件赋予版权和许可声明，有时还需要整个许可文本。
- 一些许可证还要求您发布源代码，这增加了确保您遵守许可证的工作。
- 一些许可在版权之外增加了权利和条件，例如专利、商标、数据和隐私，这可能使完全遵守许可变得更加困难。
- 更复杂的是，一些开源许可证有条件使它们与其他开源许可证不兼容。这在 GPL 许可证中最为显着。

合规性所涉及的风险不一定因其性质而更高，但可以说一些合规性要求更容易实现，因为它们需要相当少的努力，而其他一些则需要更大的努力和大量的法律分析。为了对许可证涉及的潜在合规风险进行分级，我们使用了一种交通灯分级系统。不过，重要的是要注意颜色分级代表合规性问题的估计数量和复杂性，而不是某些许可证比其他许可证风险更大——如果您了解许可证的所有合规要求并且能够满足这些要求，那么许可证无论等级如何，实际上都是无风险的。

风险等级	风险说明
红色	禁牌，合规风险高，不允许
橘色	具有重大合规风险的受限许可。只有在获得一些法律指导并根据项目或个案的基础上，才应允许此类许可，因为合规性考虑因素通常很难完全遵守。
黄色	批准的许可证，具有相当大的合规性考虑因素，例如源代码必须公开可用，并且与 Copyleft 许可证系列中的许可证一样，在不同许可证下与其他代码结合存在限制。
绿色	经批准的许可证，几乎没有合规性考虑，例如必须在代码分发中维护版权和许可声明，与 Permissive 许可证系列的大多数许可证一样。
蓝色	非 OSS/商业/专有许可

[模型参考来源]

(https://debricked.com/blog/open-source-license-families-compliance/)

10. 如何选择开源许可证?

在对现有项目做出贡献时，除非该项目使用具有不同入站许可证的贡献机制，否则通常的做法是根据整个项目所受的许可证条款做出贡献。我们将在本课程后面讨论各种贡献机制，例如开发者来源证书（DCO）和贡献者许可协议（CLA），项目可能会使用这些机制来接收代码贡献。

在参与或创建新项目时，明确要求使用代码的人（必须）做什么、允许做什么（可以）以及禁止做什么（不能）是非常重要的.选择的许可证是您指定此信息的方式。通过选择标准和常用的开源许可证，您可以帮助其他人更轻松地了解他们的权利和义务是什么。

在选择许可证时，重要的是要明确发布代码的目标。你想采用谁（什么类型的人/组织）？您想看到人们在重新分发代码时对您的代码所做的任何更改吗？您是否希望其他人能够出售您的代码以获取利润?

您还应该考虑以下常见属性：

- 发布许可、版权声明、更改摘要?
- 透露消息来源?
- 修改后的工作分配?
- 转授权?
- 私人用途还是商业用途?
- 专利授权?
- 可以使用商标吗?
- 代码可以保修吗?
- 能否承担损害赔偿责任?
- 许可范围：作为一个整体工作还是只针对特定文件?

页面上的列表包含一些常见问题，您应该在公开代码之前了解答案，并选择反映您答案的许可证。

11. 开源许可证用例

当自由软件许可证和后来的开源许可证在 80 年代末和 90 年代创建时，软件的世界很大程度上是一个物理分发的软件世界。当时起草的许可证授予被许可人使用、修改、复制等权利，条件是在分发时满足某些要求。发行意味着“当软件超出了你控制的法律界限”，无论你是在销售、出借，还是通常情况下让软件公开可用。只要你自己使用软件，你就可以(几乎)为所有许可做任何你想做的事情，但是一旦你以一种或另一种形式发布软件，就必须满足许可的条件。这给我们留下了用例的前两个区别，分布式或非分布式用例。

在起草大多数开源许可证的时候，软件在服务器或类似设备上运行并通过网络访问的做法当时还没有被广泛考虑。这种情况在 90 年代末发生了变化，特别是随着 web 服务的出现。

自由软件运动开始看到，将一个应用程序放在一个可以通过网络公开访问的服务器上的做法，是一种绕过 GPL 使源代码开放和公开可用的条件的方法——软件不是按次分发的，它只是软件执行的结果被分发。

Affero 通用公共许可证解决了这个所谓的应用程序服务提供商漏洞，该许可证旨在通过要求 AGPL 许可作品的任何网络用户都可以获得完整源代码的条款来填补这个漏洞。由于少数开源许可证也遵循了这一规定，所以我们必须将非分布

式用例分为两部分:“(I)非分布式，仅在内部使用”和“(II)非分布式，尽管作为网络服务公开可用”。

与非分布式用例一样，我们也必须将分布式用例分成两个。这源于一个单一的许可证，GPLv3，它引入了一个特定的条款，用于“设备制造商”引入了安全措施和锁，以阻止修改后的代码能够在所述设备中执行，即使代码是免费和开源的。GPLv3 的规定要求制造商建立一种机制来移除或禁用“消费类(电子)产品”中的任何锁定。因为这意味着大多数消费电子产品制造商会对他们的设备进行任何修改，导致这些设备实际上将无法销售。移动电话就是这样的情况，网络运营商不允许不安全的手机连接到他们的网络——因此设备制造商几乎无一例外地禁止了设备软件的 GPLv3 许可证。

因此，我们必须区分“(III)分布式，在通用环境(如 PC、Linux 等)中执行”和“(IV)分布式，在消费电子产品(GPLv3)中执行”的用例。

类型	使用场景
1	非分布式，仅供内部使用
2	非分布式，但作为网络服务公开可用(例如 web 或云)，“ASP 漏洞”用例
3	分布式，在通用环境下执行(如 PC、Linux)
4	在消费电子产品中分布式执行(GPLv3)

典型法律官司

【编者按】ElasticSearch 是归属 Elastic 公司的一个开源项目，提供分布式、高扩展、高实时的搜索与数据分析引擎。Amazon 公司于 2015 年基于 ElasticSearch 推出了自己的产品及服务，并将其命名为“Amazon Elasticsearch Service”，该名称使双方发生了激烈的争议。2019 年 Elastic 向 Amazon 发起了商标侵权及虚假宣

传诉讼，本文是对诉讼中 Elastic 提交的起诉状、Amazon 提交的答辩状，以及法院和解裁定的翻译。Elastic 的诉由是“商标侵权”及“虚假宣传”，即：1）Amazon 销售的产品使用“Elasticsearch”标识可能会使消费者产生混淆，或使消费者误认为 Elastic 赞助或认可 Amazon 的产品，构成了商标侵权。2）Amazon 使用“Elasticsearch”标识用于商业促销的行为，可能欺骗消费者 Amazon 提供的产品具有 Elastic 提供的“Elasticsearch”品牌产品同样的性质、特征或质量，从而影响消费者的购买决策。对此，Amazon 以合理使用等进行了抗辩。至 2022 年，该诉讼最终以双方和解告终，亚马逊网站页面以及产品服务名称中均删除了“Elasticsearch”一词，并由 Elastic 销售的 Elastic Cloud 取而代之。

和解裁定译文
案件起诉状、答辩状、和解裁决全文

The parties, having filed a Notice of Settlement on November 30, 2021 (See Docket Item No. 43), are ordered to appear before the Honorable Edward J. Davila on January 13, 2022 at 10:00 AM in Courtroom No. 4, 5th Floor, United States District Court, 280 South First Street, San Jose, California, 95113, to show cause why the case should not be dismissed pursuant to Federal Rule of Civil Procedure 41(b). On or before January 3, 2022, the parties shall file a joint statement in response to the Order to Show Cause setting forth the status of settlement efforts as well as the amount of additional time necessary to finalize and file a dismissal.

双方当事人已于 2021 年 11 月 30 日提交和解通知(见案卷号第 43 项)，并被命令于 2022 年 1 月 13 日上午 10 点在加利福尼亚州圣何塞南第一街 280 号美国地方法院 5 层第 4 号法庭，邮编 95113，在 Edward J. Davila 法官阁下面前出庭，以说明根据《联邦民事诉讼规则》第 41 条(b)款不应撤诉的理由。双方应在 2022 年 1 月 3 日或之前提交一份联合声明，对“要求说明理由的命令”做出答复，说明和解努力的情况，以及最终确定并提交撤诉所需的额外时间。

The Order to Show Cause shall be automatically vacated and the parties relieved of the obligation to file a joint statement if a stipulated dismissal pursuant to Federal Rule of Civil Procedure 41(a) is filed on or before January 3, 2022.

如果根据《联邦民事诉讼规则》第 41 条(a)款规定的撤诉在 2022 年 1 月 3 日或之前提出，则“要求说明理由的命令”应自动失效，双方解除提交联合声明的义务。

All other pretrial deadlines and hearing dates are VACATED and any pending motions are TERMINATED.

所有其他预审截止日期和听证日期均被取消，任何未决动议均被终止。

Failure to comply with any part of this Order will be deemed sufficient grounds to dismiss the action.

不遵守本命令的任何部分将被视为撤诉的充分理由。

IT IS SO ORDERED.

本案裁决如上。

翻译：刘博雅 开放原子开源基金会法律顾问

审校：

李欣博 京东集团知识产权顾问 【特邀】

申文奇 开放原子开源基金会法律顾问

郭雪雯 开放原子开源基金会法律顾问

XimpleWare 诉 Versata Software 等一审判决（2014）

【编者按】 本案中，被告 Versata 在其 DCM 软件中使用了原告 XimpleWare 根据 GPLv2.0 开源的 XML 软件中的部分代码，并将 DCM 软件销售给其客户 Ameriprise 等公司（下称“Versata 客户被告”）使用。XimpleWare 基于其拥有的与 XML 软件相关的三项专利，向法院提出的诉讼主张之一是 Versata 客户被告使用 DCM 软件的行为构成专利直接侵权。在解决此争议点时法院指出，在 Versata 客户被告

遵守 GPLv2.0 条款的前提下，GPLv2.0 明确允许其对 XML 源代码进行纯粹的使用；即使上游分发方 Versata 违反许可证条款，在 Versata 客户被告本身不违反 GPLv2.0 的情况下，其使用行为不会受到限制。若要指控 Versata 客户被告构成专利直接侵权，必须证明 Versata 客户被告在违反 GPLv2.0 条款的情况下进行了分发。但由于 XimpleWare 无法举证证明 Versata 客户被告存在分发 DCM 软件 (其中包含 XML 代码) 的行为，法院驳回了 XimpleWare 该诉讼主张。从法院的论述中可看出，单纯的“使用”行为并不触发 GPLv2.0 的条件，“分发”行为才是 GPLv2.0 条件的触发器。

判决译文节选

译文全文请浏览“阅读原文”

The Customer Defendants each move to dismiss XimpleWare's claims of direct infringement on the basis that XimpleWare has failed to plead facts showing distribution of the DCM software by any Customer Defendant.

涉案客户被告均提出动议，以 XimpleWare 未能提出事实证明任何涉案客户被告分发了 DCM 软件为由，请求驳回 XimpleWare 的直接侵权主张。

In dismissing XimpleWare's claims for direct infringement against the Customer Defendants in the FAC, the court held that regardless of the actions of the Versata Defendants, under the GPL the Customer Defendants each "retain the right to use XimpleWare's software so long as the customer does not itself breach the license by 'distributing' XimpleWare's software." Unchanged in the SAC is XimpleWare's allegation that the Customer Defendants "infringe and continue to willfully infringe the Patents by using the infringing Versata products." As use is expressly permitted under the GPL, the court's conclusion on this point is likewise unchanged: the SAC fails to state a claim for direct infringement against the Customer Defendants based on their use

of XimpleWare's code included in the DCM software.

在驳回 XimpleWare 在 FAC 中对涉案客户被告提出的直接侵权主张时，法院认为，无论涉案 Versata 被告的行为如何，根据 GPL，“只要客户本身不因‘分发’XimpleWare 的软件而违反许可证”，则涉案客户被告都“保留使用 XimpleWare 软件的权利。”XimpleWare 在 SAC 中仍然保留对涉案客户被告“通过使用侵权的 Versata 产品侵犯并持续故意侵犯专利权”的指控。由于 GPL 明确允许使用行为，法院就这一点的结论仍然没有改变：SAC 因涉案客户被告对包含在 DCM 软件中的 XimpleWare 代码使用行为而提出的直接侵权主张无法得到支持。

While use is unrestricted under the GPL, distribution is not. The GPL permits distribution only if the distributing party satisfies several specific conditions, including among other things including copy of the GPL along with the distributed program. The court previously held that XimpleWare had adequately alleged the Customer Defendants failed to satisfy the conditions for distribution of XimpleWare's software. Accordingly, the "only real issue to resolve" was "whether XimpleWare has sufficiently alleged that its software was 'distributed' by the customers when they shared the software with their independent contractors, franchisees, and producers." The court further held that XimpleWare had not, and dismissed its claims against the Customer Defendants. However, the court granted XimpleWare leave to amend to cure this defect.

虽然 GPL 不限制使用行为，但却对分发行为作出了限制。只有在分发方满足若干特定条件的情况下（包括随分发程序附上 GPL 的副本）GPL 才允许分发。法院先前认为，XimpleWare 已充分指控涉案客户被告未能满足分发 XimpleWare 软件的条件。因此，“唯一真正需要解决的问题”是“XimpleWare 是否充分指控了客户在与其独立承包商、特许经营商和制造商共享软件时‘分发’了其软件”。法院进一步认为，XimpleWare 没有就此进行充分指控，因此驳回了其对涉案客户被告的诉讼请求。不过，法院准许 XimpleWare 修正起诉状以弥补这一缺陷。

翻译：　薛杨洁

审校：　刘伟、郭雪雯

译文分享：罗盒诉玩友案一审判决（2021）

2023-06-21 20:15:00

【编者按】该案系最高人民法院发布的2021年中国法院十大知识产权案件，该案判决认定，GPLv3协议具有合同性质，是授权方和用户订立的格式化著作权协议，属于我国合同法调整的范围。此外，该案还涉及开源软件的著作权归属及权利行使、GPLv3协议“传染性”范围判定、在GPLv3协议中增加限制商业使用条款的效力，以及违反开源软件协议法律后果的多个核心开源法律问题。

第一，协议的内容具备合同特征，属于广义的合同范畴。GPLV3协议是针对某一特定的项目，并预先设定好格式化条款的协议，只要授权方选定了该协议，使用该项目的用户就必须遵守该协议，是授权方和用户之间形成的以开源软件源代码为目的的一种民事法律行为。授权方通过GPLV3协议授予不特定的用户复制、修改、再发行等权利，是设立、变更、终止民事法律权利义务关系的协议。授权方选择适用GPLV3协议传播其源代码，用户复制、修改、发行该源代码时默认承诺承继适用GPLV3协议从而保持协议的传递性，该行为是双方真实意思的表示。因此，在用户复制、修改、发行该源代码时协议成立并生效。根据我国合同法的相关规定，从理论角度对开源许可协议的成立途径进行梳理，主要为“要约说”。要约说认为，开源软件许可协议应当属于软件权利人和用户之间订立的合同，经历了正常的合同成立流程，在双方之间成立合同关系。将开源软件的发布视为发出要约，用户使用视为承诺，在用户使用开源软件时合同成立。从这一角度看开源软件许可协议应当属于广义合同的范畴。

_ **Firstly** , while its content does have features of a contract, the agreement may fall within the scope of a “contract” in a broad sense. GPLV3 is an agreement with preset formatted terms serving a specific project, and once the licensor chooses to apply this agreement, the user(s) of the project must abide by such agreement. GPLV3 is thus a

civil legal act executed between the licensor and the user(s) with the aim of opening the source of the software. The licensor grants to unspecified users, the right to copy, modify, redistribute and so on through the GPLV3, which is regarded as an agreement to establish, change, and terminate civil rights and obligations. Both the licensor's choice of applying the GPLV3 to propagate his/her source code, and the user's default commitment to subsequently apply GPLV3 when he/she copies, modifies or distributes the source code (so the agreement passes to downstream works), reflect the true intention of both parties. Therefore, the agreement is established and effective as of the moment that the user copies, modifies and distributes the source code. In accordance with relevant provisions of China's contract law, the Court examined the establishment of an open source license basically following the "offer-acceptance theory". The theory affirms that the open source license should be a contract between the owner and the user of the software, and such contractual relationship is established between the two parties with a proper execution process: while the release of an open source software is deemed as an offer, and usage by the user is deemed as a acceptance, then the contract should be established upon the user's use of the open source software. In this context, open source license should fall within the scope of "contract" in a broad sense.

第二，协议是非典型合同。与我国著作权法有关"著作权许可使用和转让合同"的规定相比较，GPLV3 协议是开源软件的作者向不特定的使用者让渡其著作权的部分人身权利和全部财产权利，权利授予的对象是不确定的，以换取使用者承诺遵守开源许可协议的许可条件和义务，如将修改后的源代码公开给社会公众共享等，开源软件许可协议并没有权利转让的对价或许可使用付酬等典型的著作权许可合同的主要条款。

Secondly, the agreement is an atypical contract. Compared with the provisions of China's Copyright Law on "contract for copyright licensing and assignment", the GPLV3 is an agreement where the author of the open source software grants his/her moral rights in part and property rights in whole of his/her copyright to an unspecified user (so the receiver of such grant is uncertain), in exchange for the user's commitment to abide by the conditions and obligations of the open source license (e.g. promise to

share his/her modified source code to the public). In an open source license, there is no consideration of right assignment or payment of royalties, etc. which is typically the main terms in a contract for copyright licensing. _

第三，协议是格式合同。GPLV3 协议是为特定开源项目开发而预先拟定，由著作权持有人向软件程序使用者提出的合同条款。GPLV3 协议序言规定，如果你发布这种程序的副本，无论以收费还是免费的模式，你必须把你获得的自由同样给予副本的接受者，你必须确保他们也能收到或得到源代码，而且你必须向他们展示这些条款以确保他们知道自己享有这样的权利。该格式化条款保持承继性，且不属于格式合同条款无效的情形，其授权内容符合我国著作权法的规定，合法有效。

_ **Thirdly** , the agreement is a format contract. The GPLV3 is the prewriting contractual provisions to develop a particular open source project, offered by the copyright holder to the user of the software program. The Preamble of the GPLV3 states that “if you distribute copies of such a program, whether gratis or for a fee, you must pass on to the recipients the same freedom that you received, and you must make sure that they, too, receive or can get the source code, and you must show them these terms so they know their rights”. As the formatted clauses maintain the inheritance and do not qualify as an invalid format contract, and the contents authorized are in accordance with the provisions of China’s Copyright Law, the GPLV3 is legally binding._

第四，对协议的承诺是通过行为作出。GPLV3 协议第 8 条规定，除非在本协议明确授权下，你不得传播或修改受保护作品。第 9 条规定，一旦修改和传播一个受保护作品，就表明你接受本协议。第 10 条规定，每当你发布一个受保护作品，其接收者自动获得来自初始授权人的授权，依照本协议可以运行、修改和传播此程序。该要约内容表明以实践行为作出承诺，无须再签订书面的合同。因此，GPLV3 协议的上述有关承诺可以用行为完成的条款符合合同法关于要约和承诺的规定，应为有效。此外，协议是通过电子文本形式由授权方或用户加入开源项目中，电子文本是一种有形的表现形式，属于以书面形式订立的合同。

_ **Fourthly** , the promise to the agreement is made by an act. The Section 8 of the GPLV3 states that “You may not propagate or modify a covered work except as expressly

provided under this License." Section 9 states that "You may not propagate or modify a covered work except as expressly provided under this License." Section 10 states that "Each time you convey a covered work, the recipient automatically receives a license from the original licensors, to run, modify and propagate that work, subject to this License". The content of this offer indicates that a promise is made by a practical act, and no further contract signing required. Therefore, the above-mentioned provisions of the GPLV3 that a promise can be completed by an act are in compliance with the provisions of contract law on "offer-acceptance theory", and thus shall be valid. Furthermore, the licensor or user adds the agreement to an open source project in the form of electronic text, which is a tangible form of expression and thus is regarded as a contract executed in writing.

综上，GPLV3 协议具有合同性质，是授权方和用户订立的格式化著作权协议，属于我国合同法调整的范围。

In conclusion, the GPLV3 is by nature contractual and is a formatted copyright agreement concluded between the licensor and the user, which falls within the scope of the adjustment of China's contract law.

译文分享：SCO 诉 IBM 上诉判决（2018）

【编者按】本案是曾被 Linux 视为"生存威胁"的诉讼案，历时 18 年最终以 1425 万美元达成和解。本案起因于 SCO 前身与 IBM 之间就操作系统达成的合作项目 Monterey。SCO 指控称，IBM 将属于 SCO 的知识产权贡献给了 Linux 开源社区并创建了 UNIX 操作系统 AIX，进而使这些源代码以 GPL 许可证形式免费被使用。2018 年，美国上诉法院撤销了基于代码盗用主张对 IBM 的简易判决并发回地区法院重审，并就对 IBM 侵权性干扰的主张维持原判，对驳回 SCO 修改诉状的请求维持原判。2021 年，随着 SCO 宣告破产，双方最终和解。

判决书译文节选

译文全文请点击下方“阅读原文”

2. 通过向 Linux 不正当的披露进行间接干扰

地区法院认为，根据犹他州的法律，SCO 基于 IBM 公司向 Linux 披露的间接干扰理论并不可诉。SCO 声称，IBM 错误地向开源社区披露专有的 UNIX 素材，导致 SCO 的潜在客户放弃其收费产品，转向免费的替代产品，也导致 SCO 在 UNIX-on- Intel 市场上的销售额大幅下降。换句话说，这一理论是基于被告被指控的基于市场的活动，这些活动可能导致潜在客户不再与原告做生意。犹他州法院是否会将这种理论认可为侵权干扰主张，则是州法律中尚未解决的问题。

“当一项上诉提出的是州法律尚未解决的问题时，我们通常必须‘试图预测最高法院将如何解释该问题’”。“此外，‘如果没有最高法院的明确指导，我们一般不会扩大州法律’”。事实上，我们已经指出，鼓励联邦法院适用州法律下的新的法律原则的一方必须能“有力地证明”如果提出该问题，该州的最高法院将会采用这个原则。考虑到这一点，我们适用犹他州的法律。

犹他州法律明确接受故意干扰理论，该理论基于潜在或预期的客户——而非仅是现有客户——由于被告的不正当干扰行为而流失的证据。

此外，犹他州法律并不要求被告为诱使第三方切断与原告的经济关系而去直接面对第三方。犹他州最高法院在 Leigh Furniture & Carpet Co. v. Isom 一案的开创性的判决中说明了这一点。在该案中，被告的代表经常在“营业时间内频繁造访原告的商店，与他面对面，询问他，并就其业务开展方式提出要求与询问。”这些破坏性的侵扰“一再地干扰了销售活动，导致[原告]的顾客有抱怨和投诉，并不止一次导致顾客离开商店”。尽管被告与被赶走的第三方客户之间没有直接的交流，但犹他州最高法院仍然认为这种干扰行为是可由被告起诉的。

可以肯定的是，Leigh Furniture 案支持了基于被告的对未直接接触潜在客户的侵权性干扰的主张，但 Leigh Furniture 案并没有明确表示，任何对公众和一般市场的不正当行为，只要竞争对手的商业关系因不正当行为而受到损害，就可以构成侵权性干扰的主张。实际上，这会将几乎所有基于市场的非法活动的主张转化为侵权性干扰的主张，在犹他州的判例法中，我们并没有发现任何迹象表明州最高法院会接受这样一种扩大性的解释。至少，由于 SCO 没有就这一点引用任何控制性的案例，因此未能“有力地证明”犹他州最高法院在遇到这个问题时会证实

该理论。因此，我们同意地区法院的观点，即 SCO 的间接干扰理论在犹他州法律下并不可诉。

翻译 &交叉审校：郭雪雯、张苏兵、刘博雅、陈文 **审校**：王荷舒