<uses-sdk>

本文内容

语法：

```
<uses-sdk android:minSdkVersion="integer"
          android:targetSdkVersion="integer"
          android:maxSdkVersion="integer" />
```

Google Play 过滤

Google Play 利用在您的应用清单中声明的 `<uses-sdk>` 属性从不符合其平台版本要求的设备上滤除您的应用。 在设置这些属性前，确保您了解 Google Play 过滤器。

包含它的文件：

`<manifest>`

说明：

您可以通过整数形式的 API 级别表示应用与一个或多个版本的 Android 平台的兼容性。 应用表示的 API 级别将与给定 Android 系统的 API 级别进行比较，其结果在不同 Android 设备上可能有所差异。

尽管名称是这样，但该元素实际上是用于指定 API 级别，*而不是* SDK（软件开发工具包）或 Android 平台的版本号。API 级别始终是单个整数。 您无法通过 API 级别关联的 Android 版本号来推知 API 级别（例如，它不同于主版本号，也不同于主版本号与次版本号之和）。

另请阅读有关对您的应用进行版本控制的文档。

属性：

`android:minSdkVersion`

一个用于指定应用运行所需最低 API 级别的整数。 如果系统的 API 级别低于该属性中指定的值，Android 系统将阻止用户安装应用。 您应该始终声明该属性。

> **注意**：如果您不声明该属性，系统将假定默认值为“1”，这表示您的应用兼容所有 Android 版本。 如果您的应用*并不*兼容所有版本（例如，它使用 API 级别 3 中引入的 API），并且您尚未声明正确的 `minSdkVersion`，则当应用安装在 API 级别小于 3 的系统上时，应用将在运行时尝试访问不可用的 API 时发生崩溃。 因此，请务必在 `minSdkVersion` 属性中声明合适的 API 级别。

`android:targetSdkVersion`

一个用于指定应用的目标 API 级别的整数。如果未设置，其默认值与为 `minSdkVersion` 指定的值相等。

该属性用于通知系统，您已针对目标版本进行测试，并且系统不应启用任何兼容性行为来保持您的应用与目标版本的向前兼容性。 应用仍可在较低版本上运行（最低版本为 `minSdkVersion`）。

在 Android 随着每个新版本的推出而进化的过程中，一些行为甚至是外观可能会发生变化。不过，如果平台的 API 级别高于您的应用的 `targetSdkVersion` 所声明的版本，系统就可以通过启用兼容性行为来确保您的应用继续以您所期望的方式工作。 您可以通过将 `targetSdkVersion` 指定为与应用所运行平台的 API 级别一致来停用此类兼容性行为。 例如，如果将该值设置为“11”或更高，系统便可在您的应用运行在 Android 3.0 或更高版本的平台上时对其应用新的默认主题 (Holo)，还可在您的应用运行在更大屏幕上时停用屏幕兼容性模式（因为对 API 级别 11 的支持隐含了对更大屏幕的支持）。

系统可根据您为该属性设置的值启用许多兼容性行为。 `Build.VERSION_CODES` 参考资料中的相应平台版本对其中的几种行为做了说明。

要让您的应用与各 Android 版本保持同步，您应该增加该属性的值，使其与最新 API 级别一致，然后在相应平台版本上对您的应用进行全面测试。

引入的版本：API 级别 4

`android:maxSdkVersion`

一个指定作为应用设计运行目标的最高 API 级别的整数。

在 Android 1.5、1.6、2.0 及 2.0.1 中，系统会在安装应用以及系统更新后重新验证应用时检查该属性的值。 在任一情况下，如果应用的 `maxSdkVersion` 属性低于系统本身使用的 API 级别，系统均不允许安装应用。 在系统更新后重新验证这种情况下，这实际上相当于将您的应用从设备中移除。

为说明该属性在系统更新后对您的应用的影响，请看看下面这个示例：

Google Play 上发布了一个在其清单中声明了 `maxSdkVersion="5"` 的应用。 一位设备运行 Android 1.6（API 级别 4）的用户下载并安装了该应用。 几周后，该用户收到了 Android 2.0（API 级别 5）OTA 系统更新。 更新安装后，系统检查该应用的 `maxSdkVersion` 并顺利完成了对其的重新验证。 应用仍可照常工作。 不过，一段时间后，设备又收到了一个系统更新，这次是更新到 Android 2.0.1（API 级别 6）。 更新完成后，系统无法再重新验证应用，因为此时系统本身的 API 级别 (6) 已超过该应用支持的最高级别 (5)。 系统会使该应用对用户不可见，这实际上相当于将它从设备上删除。

> **警告：**不建议声明该属性。 首先，没有必要设置该属性，将其作为阻止您的应用部署到 Android 平台新发布版本上的一种手段。 从设计上讲，新版本平台完全向后兼容。 只要您的应用只使用标准 API 并遵循部署最佳实践，应该能够在新版本平台上正常工作。 其次，请注意在某些情况下，声明该属性可能**导致您的应用在系统更新至更高 API 级别后被从用户设备中移除**。大多数可能安装您的应用的设备都会定期收到 OTA 系统更新，因此您应该在设置该属性前考虑这些更新对您的应用的影响。

引入的版本：API 级别 4

未来的 Android 版本（Android 2.0.1 之后的版本）在安装或重新验证时将不再检查或强制执行 `maxSdkVersion` 属性。 不过，Google Play 在向用户呈现可供下载的应用时将继续使用该属性作为过滤器。

引入的版本：

API 级别 1

什么是 API 级别？

API 级别是一个对 Android 平台版本提供的框架 API 修订版进行唯一标识的整数值。

Android 平台提供了一种框架 API，应用可利用它与底层 Android 系统进行交互。 该框架 API 由以下部分组成：

- 一组核心软件包和类
- 一组用于声明清单文件的 XML 元素和属性
- 一组用于声明和访问资源的 XML 元素和属性
- 一组 Intent
- 一组应用可请求的权限，以及系统中包括的权限强制执行。

每个后续版本的 Android 平台均可包括对其提供的 Android 应用框架 API 的更新。

框架 API 更新的设计用途是使新 API 与早期版本的 API 保持兼容。 也就是说，大多数 API 更改都是新增更改，会引入新功能或替代功能。 在 API 的某些部分得到升级时，旧版的被替换部分将被弃用，但不会被移除，这样现有应用仍可使用它们。 在极少数情况下，可能会修改或移除 API 的某些部分，但通常只有在为了确保 API 稳健性以及应用或系统安全性时，才需要进行此类更改。 所有其他来自早期修订版的 API 部分都将结转，不做任何修改。

Android 平台提供的框架 API 使用叫做“API 级别”的整数标识符指定。 每个 Android 平台版本恰好支持一个 API 级别，但隐含了对所有早期 API 级别（低至 API 级别 1）的支持。 Android 平台初始版本提供的是 API 级别 1，后续版本的 API 级别递增。

下表列出了各 Android 平台版本支持的 API 级别。 如需了解有关运行各版本的设备的相对数量的信息，请参阅“平台版本”信息中心页面。

平台版本	API 级别	VERSION_CODE	备注
Android 7.0	24	N	平台亮点
Android 6.0	23	M	平台亮点
Android 5.1	22	LOLLIPOP_MR1	平台亮点
Android 5.0	21	LOLLIPOP	
Android 4.4W	20	KITKAT_WATCH	仅限 KitKat for Wearables
Android 4.4	19	KITKAT	平台亮点
Android 4.3	18	JELLY_BEAN_MR2	平台亮点
Android 4.2、4.2.2	17	JELLY_BEAN_MR1	平台亮点
Android 4.1、4.1.1	16	JELLY_BEAN	平台亮点
Android 4.0.3、4.0.4	15	ICE_CREAM_SANDWICH_MR1	平台亮点
Android 4.0、4.0.1、4.0.2	14	ICE_CREAM_SANDWICH	
Android 3.2	13	HONEYCOMB_MR2	
Android 3.1.x	12	HONEYCOMB_MR1	平台亮点
Android 3.0.x	11	HONEYCOMB	平台亮点
Android 2.3.4 Android 2.3.3	10	GINGERBREAD_MR1	平台亮点
Android 2.3.2 Android 2.3.1 Android 2.3	9	GINGERBREAD	
Android 2.2.x	8	FROYO	平台亮点
Android 2.1.x	7	ECLAIR_MR1	平台亮点
Android 2.0.1	6	ECLAIR_0_1	
Android 2.0	5	ECLAIR	
Android 1.6	4	DONUT	平台亮点
Android 1.5	3	CUPCAKE	平台亮点
Android 1.1	2	BASE_1_1	
Android 1.0	1	BASE	

API 级别在 Android 中的使用

API 级别标识符在确保尽可能为用户和应用开发者提供最佳体验方面发挥着重要作用：

- 它允许 Android 平台描述其支持的最高框架 API 修订版

- 它允许应用描述其需要的框架 API 修订版
- 它允许系统协商应用在用户设备上的安装，从而不安装不兼容版本的应用。

每个 Android 平台版本都将其 API 级别标识符存储在 Android 系统自身内部。

应用可以利用框架 API 提供的清单元素 (`<uses-sdk>`) 来说明其可以运行的最低和最高 API 级别，以及其在设计上支持的首选 API 级别。 该元素具有以下三个重要属性：

- `android:minSdkVersion` - 指定能够运行应用的最低 API 级别。 默认值为“1”。
- `android:targetSdkVersion` - 指定运行应用的目标 API 级别。 在某些情况下，这允许应用使用在目标 API 级别中定义的清单文件元素或行为，而不是仅限于使用那些针对最低 API 级别定义的元素。
- `android:maxSdkVersion` - 指定能够运行应用的最高 API 级别。 **重要说明：**在使用该属性之前，请先阅读 `<uses-sdk>` 文档。

例如，要指定应用运行所需的最低系统 API 级别，应用需要在其清单中加入一个带 `android:minSdkVersion` 属性的 `<uses-sdk>` 元素。`android:minSdkVersion` 是一个整数值，对应于能够运行应用的最低版本 Android 平台的 API 级别。

当用户试图安装应用，或在系统更新后重新验证应用时，Android 系统会先检查应用清单中的 `<uses-sdk>` 属性，然后将这些属性的值与其自己的内部 API 级别进行对比。 只有在符合以下条件时，系统才允许安装开始：

- 如果声明了 `android:minSdkVersion` 属性，其值必须小于或等于系统的 API 级别整数。 如果未声明，则系统假定应用需要 API 级别 1。
- 如果声明了 `android:maxSdkVersion` 属性，其值必须大于或等于系统的 API 级别整数。如果未声明，则系统假定应用没有最高 API 级别。 如需了解有关系统如何处理该属性的详细信息，请阅读 `<uses-sdk>` 文档。

如果在应用清单中进行了声明，`<uses-sdk>` 元素的内容可能如下所示：

```
<manifest>
  <uses-sdk android:minSdkVersion="5" />
  ...
</manifest>
```

应用在 `android:minSdkVersion` 中声明 API 级别的主要原因是，告知 Android 系统，它使用了在指定 API 级别 *引入*的 API。 如果由于某种原因将应用安装在 API 级别更低的平台上，则它会在运行时试图访问不存在的 API 时发生崩溃。系统通过在应用需要的最低 API 级别高于目标设备上的平台版本时不允许安装应用来防止出现这种结果。

例如，`android.appwidget` 软件包是随 API 级别 3 引入的。如果应用使用该 API，则必须使用“3”一值声明 `android:minSdkVersion` 属性。随后，应用便可安装在 Android 1.5（API 级别 3）和 Android 1.6（API 级别 4）等平台上，但不能安装在 Android 1.1（API 级别 2）和 Android 1.0（API 级别 1）平台上。

如需了解有关如何指定应用 API 级别要求的详细信息，请参阅清单文件文档的 `<uses-sdk>` 部分。

开发注意事项

下文提供有关您在开发应用时应该考虑的 API 级别的信息。

应用向前兼容性

Android 应用一般向前兼容新版本的 Android 平台。

因为几乎所有对框架 API 的更改都是新增更改，所以使用 API 任何给定版本（其 API 级别所指定版本）开发的 Android 应用均向前兼容更新版本的 Android 平台以及更高 API 级别。 应用应该能够在所有后期版本的 Android 平台上运行，除非在个别情况下应用使用的某个 API 部分后来由于某种原因被移除。

向前兼容性很重要，因为许多 Android 设备都接收 OTA 系统更新。 用户可以安装您的应用并顺利使用，然后通过接收 OTA 更新升级到新版本的 Android 平台。 安装更新后，您的应用将运行在新运行时版本的环境中，但这个版本具有您的应用所依赖的 API 和系统功能。

在某些情况下，在该 API *之下*所做的更改（例如对底层系统本身的更改）可能会影响运行在新环境中的应用。 因此，作为应用开发者，您必须了解应用在各系统环境中的外观和行为。 为帮助您在各种版本的 Android 平台上测试您的应用，Android SDK 提供了多个可供您下载的平台。每个平台都包括一个兼容的系统映像，您可以在 AVD 中运行该映像来测试您的应用。

应用向后兼容性

Android 应用不一定向后兼容比其编译时所针对的目标版本更久远的 Android 平台版本。

每个新版本的 Android 平台都可能包括新的框架 API，例如那些能够让应用使用新的平台功能或者替换现有 API 部分的 API。 应用可以在运行于新平台时使用这些新 API，如上所述，也可以在运行于更新版本的平台（API 级别所指定的平台）上时使用这些新 API。 反之，由于早期版本的平台不包括新 API，因此使用新 API 的应用无法运行在这些平台上。

尽管 Android 设备降级至之前版本平台的情况不太可能发生，但必须意识到，在现实情况下可能有许多设备运行的是早期版本的平台。 即便是在能够获得 OTA 更新的设备中，也可能发生一些设备获得的更新滞后，可能在相当长时间内无法获得更新的情况。

选择平台版本和 API 级别

当您开发应用时，需要选择您所编译的应用所针对的目标平台版本。 一般而言，您编译的应用所针对的目标版本应该是您的应用所能支持的最低平台版本。

您可以通过在编译应用时依次降低其所针对的目标编译版本来确定可能的最低平台版本。 确定最低版本后，您应该使用相应平台版本（和 API 级别）创建一个 AVD，然后对您的应用进行全面测试。 请务必在应用的清单中声明 `android:minSdkVersion` 属性，并将其值设置为平台版本的 API 级别。

声明最低 API 级别

如果您构建的应用使用的 API 或系统功能是在最新平台版本中引入的，则应将 `android:minSdkVersion` 属性的值设置为最新平台版本的 API 级别。 这样做可确保用户只能将您的应用安装在运行兼容版本 Android 平台的设备上， 并进而确保您的应用可以在用户设备上正常工作。

如果您的应用使用在最新平台版本中引入的 API，但*未*声明 `android:minSdkVersion` 属性，则应用可以在运行最新版本平台的设备上正常工作，但*无法*在运行早期版本平台的设备上正常工作。 在后一种情况下，应用将在运行时试图使用早期版本上不存在的 API 时发生崩溃。

针对更高 API 级别进行测试

编译应用之后，您应该确保在应用的 `android:minSdkVersion` 属性所指定的平台上对其进行测试。 为此，请使用您的应用所需的平台版本创建一个 AVD。 此外，为确保向前兼容性，您还应在所有使用的 API 级别高于您的应用所用 API 级别的平台上运行并测试您的应用。

Android SDK 提供了多个可供您使用的平台版本（包括最新版本），并提供了可供您在必要时下载其他平台版本的更新器工具。

要访问该更新器，请使用位于 <sdk>/tools 目录的 `android` 命令行工具。 您可以通过执行 `android sdk` 来启动 SDK 更新器。 您还可以直接双击 android.bat (Windows) 或 Android (OS X/Linux) 文件。

要在模拟器中于不同平台版本上运行您的应用，请为您想测试的每个平台版本创建一个 AVD。 如需了解有关 AVD 的详细信息，请参阅 创建和管理虚拟设备。 如果您要使用物理设备进行测试，请确保您知晓它所运行的 Android 平台的 API 级别。 请参阅本文顶部表格中所列的平台版本及其 API 级别。

按 API 级别过滤参考文档

Android Developers 网站在每个参考文档页面的右上角提供了一个“Filter by API Level”控件。 您可以利用该控件，根据应用在其清单文件的 `android:minSdkVersion` 属性中指定的 API 级别，只显示您的应用实际可以访问的 API 部分的对应文档。

要使用过滤，请选择用于启用过滤的复选框，该复选框就在页面搜索框下方。 然后将“Filter by API Level”控件设置为您的应用所指定的同一 API 级别。 请注意，后期 API 级别中引入的 API 随即灰显，并且其内容会被屏蔽，因为您的应用无法访问它们。

在文档中按 API 级别进行过滤并不能让您查看每个 API 级别新增或引入的 API - 它仅仅是提供了一种途径，让您能够查看与给定 API 级别关联的整个 API，同时将后期 API 级别中引入的 API 元素排除在外。

如果您决定不想过滤 API 文档，只需使用复选框停用该功能。 默认情况下系统会停用 API 级别过滤，这样无论 API 级别如何，您都能查看完整的框架 API。

还请注意，各 API 元素的参考文档指定的是引入各元素的 API 级别。 软件包和类的 API 级别在各文档页面内容区域的右上角以“Since <api 级别>”形式指定。 类成员的 API 级别在其位于右边距的详细说明标头中指定。