连接安卓设备

您可以使用USB电缆将Android设备连接到开发PC，以构建，运行，调试和分析Qt Creator的应用程序。使用 Qt 5 进行开发时支持 Android 版本 4.1（API 级别 16）或更高版本的设备，使用 Qt 6 进行开发时支持具有 Android 版本 6.0（API 级别 23）的设备。

要针对 Android 进行开发，您必须具有一个工具链，用于在开发 PC 上安装用于为 Android 设备构建应用程序。Qt Creator可以自动下载并安装工具链，并创建一个合适的构建和运行工具包，其中包含工具链和适用于设备架构的Android Qt版本。

从 Qt 5.14.0 开始，适用于安卓的 Qt 软件包包含作为一个整体安装的所有架构 （ABI）。

要为 Java 启用有用的代码编辑功能（如代码完成、突出显示、函数工具提示和在代码中导航），请添加 Java 语言服务器。

安卓调试桥（adb）命令行工具已集成到Qt Creator中，使您能够将应用程序部署到连接的安卓设备，运行它们并读取其日志。它包括在开发主机上运行的客户端和服务器，以及在模拟器或设备上运行的守护程序。

要求

要使用Qt Creator开发适用于Android的Qt应用程序，您需要适用于Android 5.2或更高版本的Qt，以及Qt Creator可以自动下载，安装和配置的工具链。有关详细信息，请参阅手动安装必备组件。

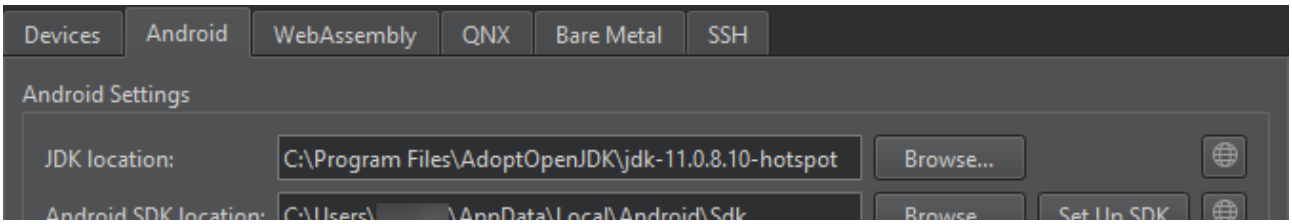
指定安卓设备设置

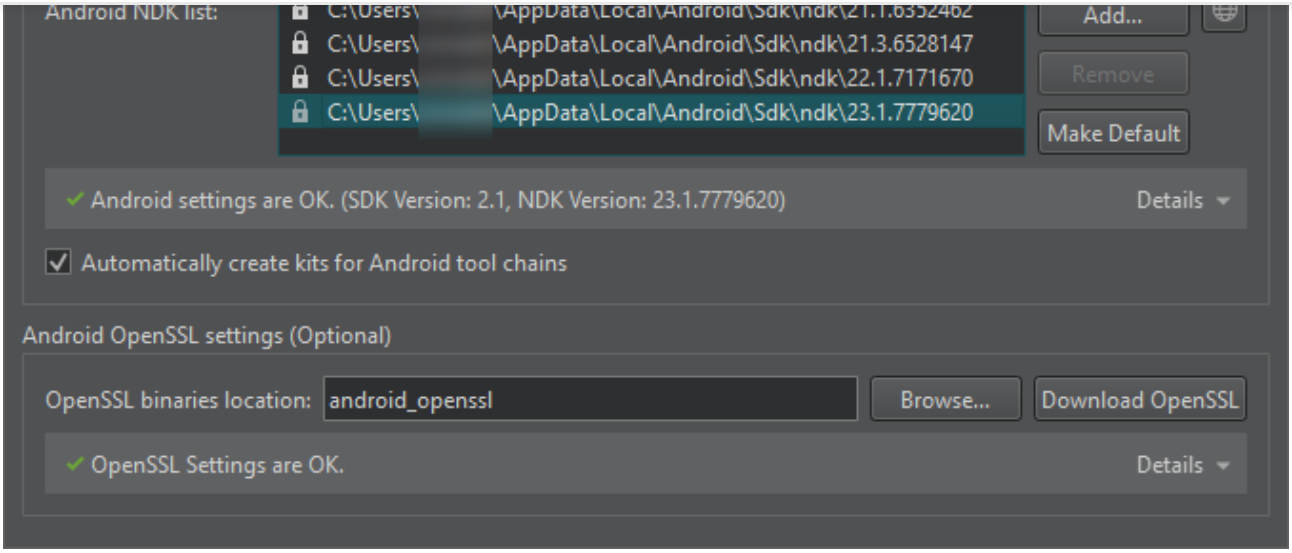
Qt Creator提供自动安装所有必要的软件包和工具，并通过创建调试器，工具链和工具包来设置您的开发环境。您可以使用Qt创建者来：

- 下载并解压缩安卓 SDK 命令行工具。
- 安装或更新基本软件包，如 NDK、构建工具和平台工具。

要设置安卓系统的开发环境：

1. 在 Windows 和 Linux 上选择"**>设备>安卓**设备编辑>**首选项**"，或在 macOS 上选择"设备>**安卓设备**"> **Qt 创建者**>首选项"。





2. 在**"JDK 位置"**字段中，设置 JDK 的路径。Qt 创建器检查 JDK 安装并报告错误。

   默认情况下，Qt 创建者会尝试查找受支持的领养打开JDK 或开放JDK 安装。如果未找到任何路径，则必须手动设置路径。如果未安装受支持的 JDK，请选择以在默认浏览器中打开 JDK 下载网页。

   > **注意：** 我们建议使用 64 位 JDK，因为 32 位 JDK 可能会导致 出现问题，并且某些包可能不会列出。`cmdline-tools`

3. 在**"安卓 SDK 位置"**字段中，设置要在其中安装安卓 SDK 命令行工具的文件夹的路径。
4. 选择**"设置开发工具包"**以自动下载 Android SDK 命令行工具并将其解压缩到所选路径。

   SDK 管理器会检查工具链是否已安装。如果程序包丢失或需要更新，SDK 管理器会提供添加或删除这些程序包的功能。在采取行动之前，它会提示您接受即将进行的更改。此外，它还会根据需要提示您接受 Google 许可证。

5. 已安装的 NDK 版本列在**安卓 NDK 列表中**。锁定的项目由 SDK 管理器安装，只能从 **Android SDK 管理器**对话框中进行修改。有关更多信息，请参阅 管理安卓 NDK 软件包。
6. 选中**"自动为 Android 创建工具包"工具链**复选框，以允许 Qt 创建器为您创建工具包。Qt创建者如果找不到合适的Qt版本，则会显示警告。
7. （可选）在**"安卓开放SSL 设置"**组中，设置预构建的开放SSL 库的路径。

   对于需要支持开放SSL的Qt应用程序，Qt创建者允许快速将安卓开放SSL支持添加到您的项目中。有关详细信息，请参阅添加外部库。

8. Select **Download OpenSSL** to download the OpenSSL repository to the selected path. If the automatic download fails, the download web page opens for manual download.

Manual Setup

> **Note:** We recommend that you use the latest Android SDK Command-Line Tools. Using Android SDK Tools version 25.2.5 or earlier is not supported because they cannot be fully integrated with Qt Creator.

However, if the automatic setup does not meet your needs, you can download and install Android SDK Command-line Tools, and then install or update the NDKs, tools and packages needed for development. For more information, see Getting Started with Qt for Android.

The Android SDK Command-Line Tools download URL, the essential packages list, and the appropriate NDK for each Qt version are defined in a JSON configuration file. The file is located under the user's Qt Creator resource folder:

```
# Linux and macOS
~/.config/QtProject/qtcreator/android/sdk_definitions.json

# Windows
C:\Users\Username\AppData\Local\QtProject\qtcreator\android\sdk_definitions.json
```

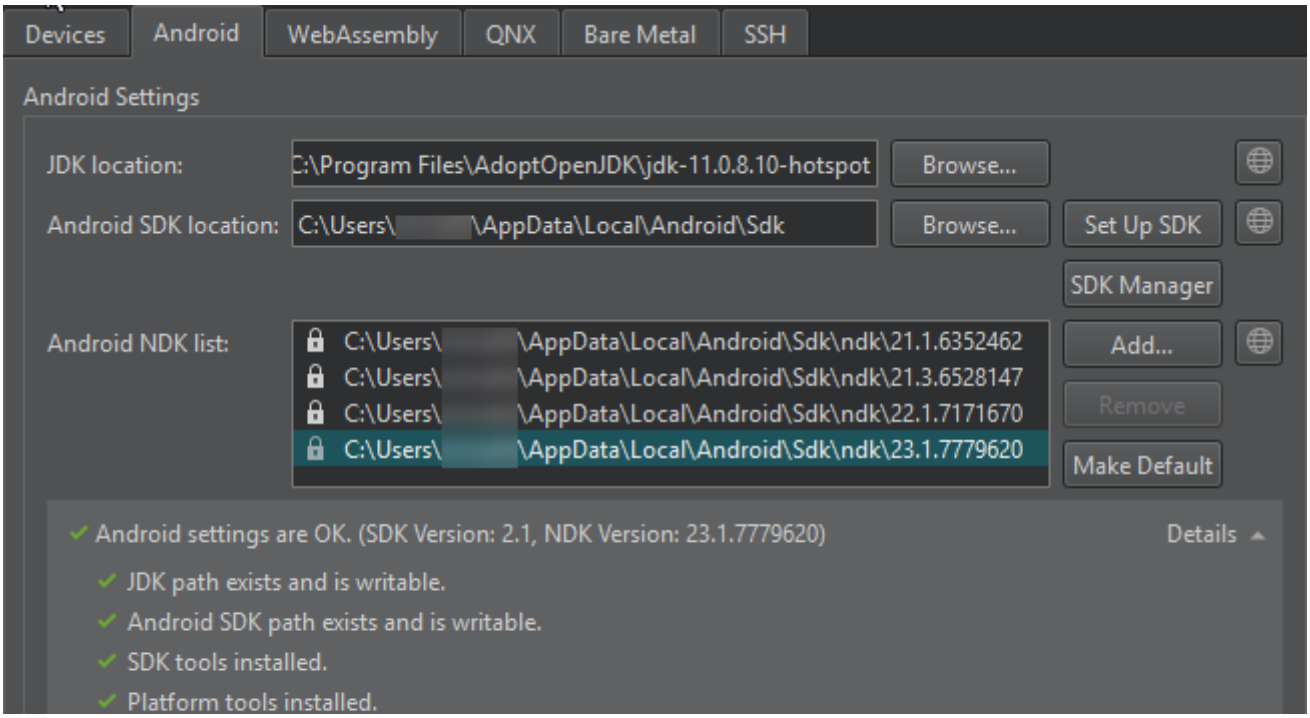
For example, the SDK configuration file defines the NDK version 19.2.5345600 to be used for Qt 5.12.0 to 5.12.5 and Qt 5.13.0 to 5.13.1 versions:

```
"specific_qt_versions": [
    {
        "versions": ["5.12.[0-5]", "5.13.[0-1]"],
        "sdk_essential_packages": ["build-tools;28.0.2", "ndk;19.2.5345600"],
        "ndk_path": "ndk/19.2.5345600"
    }
]
```

You can view the latest version of the configuration file that is up-to-date with the Android SDK and NDK changes, sdk_definitions.json, in Git.

Managing Android NDK Packages

To view the installed Android NDK versions, select **Edit** > **Preferences** > **Devices** > **Android** on Windows and Linux or **Qt Creator** > **Preferences** > **Devices** > **Android** on macOS.



The locked versions were installed by the SDK Manager, and can only be modified from the **Android SDK Manager** dialog. For more information, see Managing Android SDK Packages.

To manually download NDKs, select .

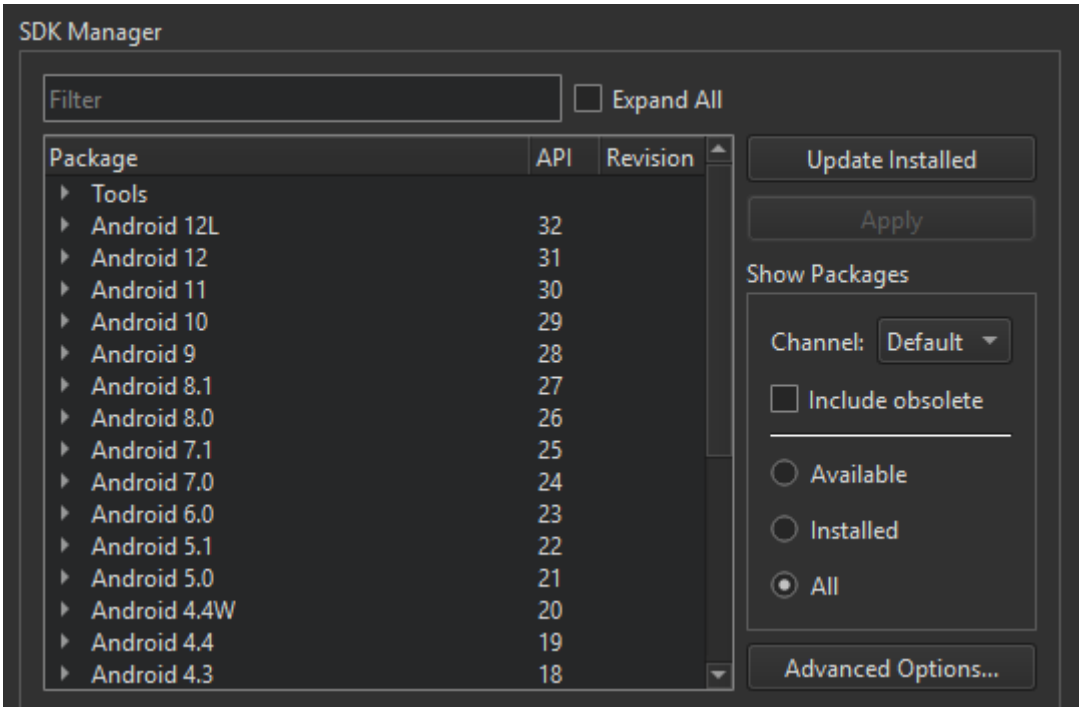
To use the selected NDK version for all Qt versions by default, select **Make Default**.

To add custom NDK paths manually to the global list of NDKs, select **Add**. This creates custom tool chains and debuggers associated to that NDK. However, you have to manually create a kit that uses the custom NDK. For more information, see Adding Kits.

Managing Android SDK Packages

Since Android SDK Tools version 25.3.0, only a command-line tool, sdkmanager, is provided by Android for SDK package management. To make SDK management easier, Qt Creator provides an SDK Manager for installing, updating, and removing SDK packages. You can still use sdkmanager for advanced SDK management.

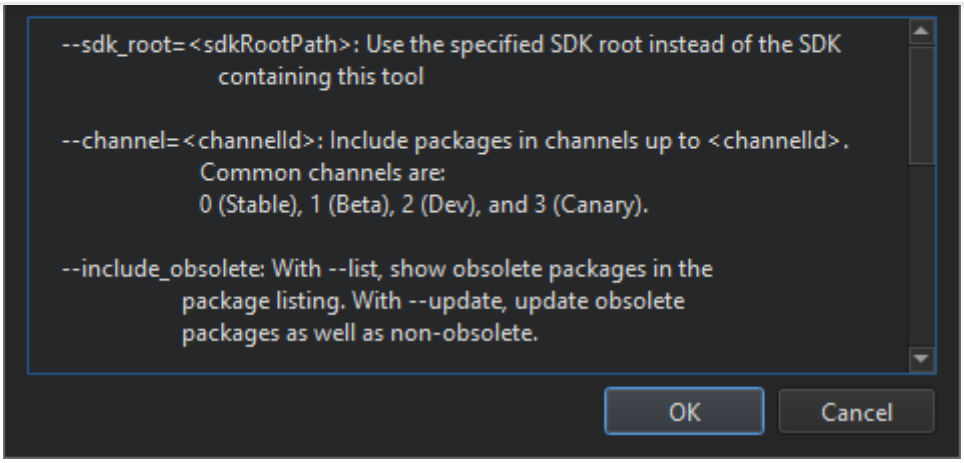
To view the installed Android SDK packages, select **Edit** > **Preferences** > **Devices** > **Android** > **SDK Manager** on Windows and Linux or **Qt Creator** > **Preferences** > **Devices** > **Android** > **SDK Manager** on macOS.



You can show packages for the release channel you select in **Show Packages** > **Channel**. Common channel IDs include **Stable**, **Beta**, **Dev**, and **Canary**. To show and update also obsolete packages, select **Include obsolete**. To filter packages, select **Available**, **Installed**, or **All**.

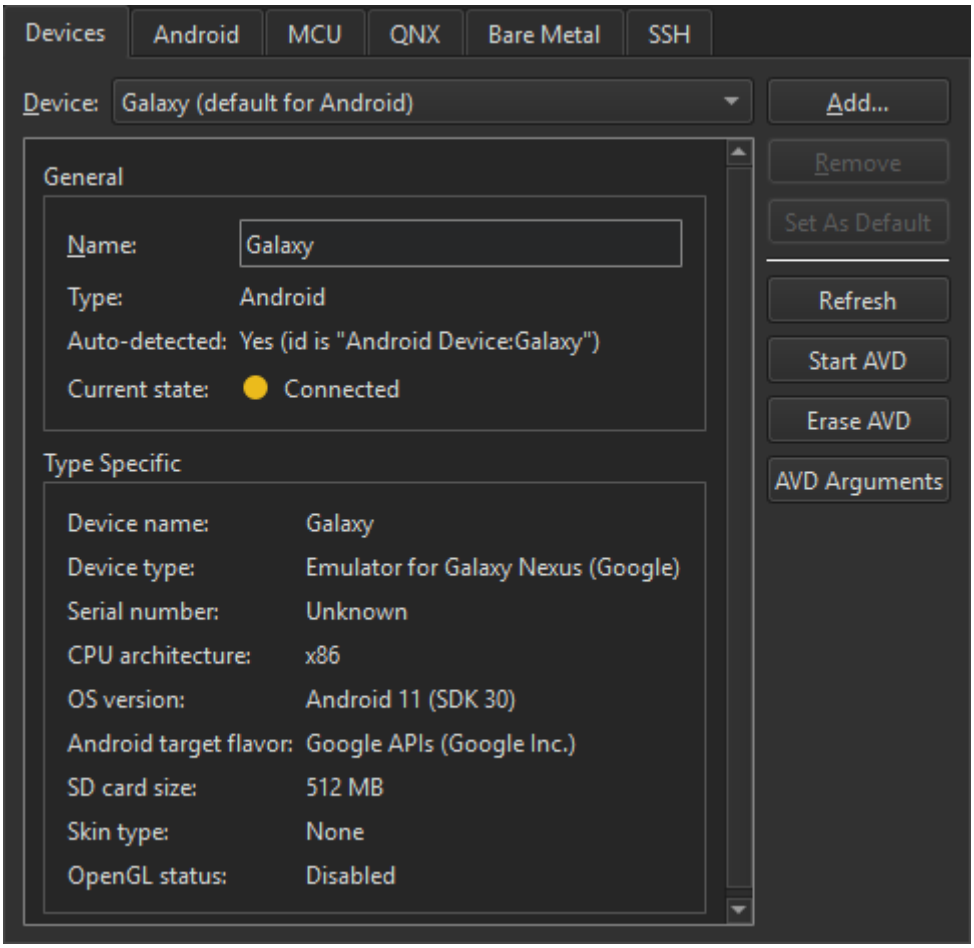
To update the installed Android SDK packages, select **Update Installed**. Select the packages to update, and then select **Apply**.

To specify advanced sdkmanager settings, select **Advanced Options** and enter arguments in the **SDK Manager arguments** field. The available arguments are listed and described in **Available arguments**.



Managing Android Virtual Devices (AVD)

The available AVDs are listed in **Edit** > **Preferences** > **Devices** on Windows and Linux or **Qt Creator** > **Preferences** > **Devices** > on macOS. You can add more AVDs.

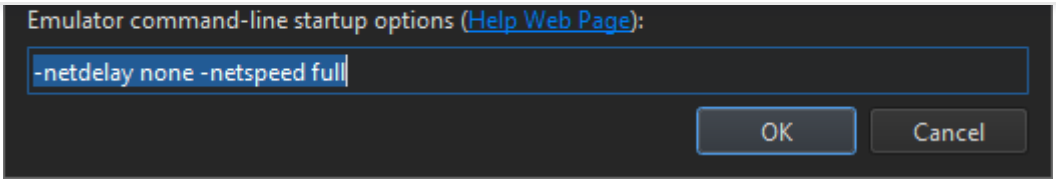


You can see the status of the selected device in **Current state**. To update the status information, select **Refresh**.

To start an AVD, select **Start AVD**. Usually, you don't need to start AVDs separately because they are automatically started when you select them in the kit selector to deploy applications to them.

To remove an AVD from the list and the kit selector, select **Erase AVD**.

To specify options for starting an AVD, select **AVD Arguments**.



Specify the options in **Emulator command-line startup options**. For available options, see Start the emulator from the command line.

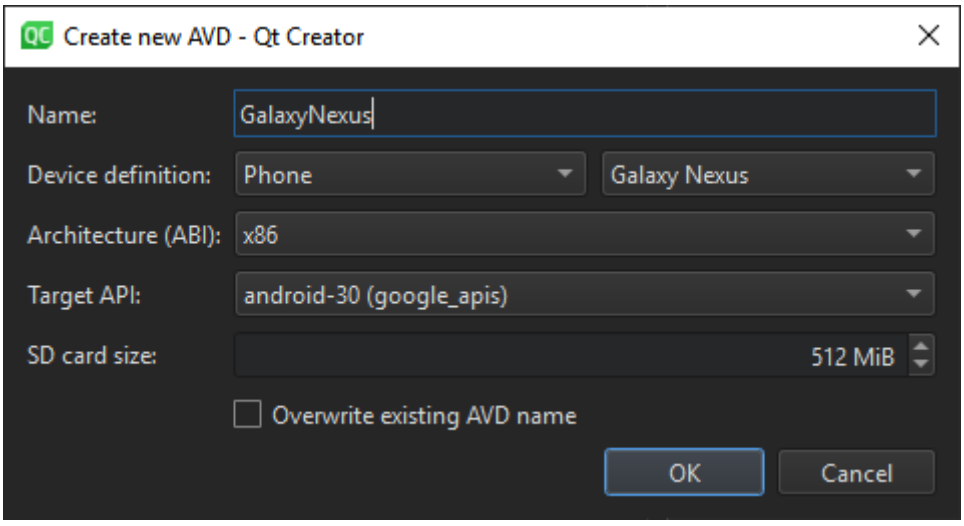
> **Note:** The Android Emulator has a bug that prevents it from starting on some systems. If an AVD does not start, you can try starting it manually by running the following commands:

```
cd <ANDROID_SDK>/emulator
./emulator -avd <AVD_NAME>
```

Creating a New AVD

To create new virtual devices:

1. Select **Edit** > **Preferences** > **Devices** > **Add** > **Android Device** on Windows and Linux or **Qt Creator** > **Preferences** > **Devices** > **Add** > **Android Device** on macOS to open the **Create New AVD** dialog.



2. Set the name, definition, architecture, target API level, and SD card size of the device.
3. Select **OK** to create the AVD.

For more advanced options for creating a new AVD, use the command-line tool avdmanager or the Android Studio's native AVD Manager UI.

Debugging on Android Devices

Debugging is enabled in different ways on different Android devices. Look for **USB Debugging** under **Developer Options**. On some devices **Developer Options** is hidden and becomes visible only when you tap the **Build number** field in **Settings** > **About** several times. For more information, see Configure on-device developer options.

Note: Qt Creator cannot debug applications on Android devices if Android Studio is running. If the following message is displayed in Application Output, close Android Studio and try again:

```
Ignoring second debugger -accepting and dropping.
```

‹ Connecting Devices

Connecting Bare Metal Devices ›

Qt DOCUMENTATION

Feedback Sign In