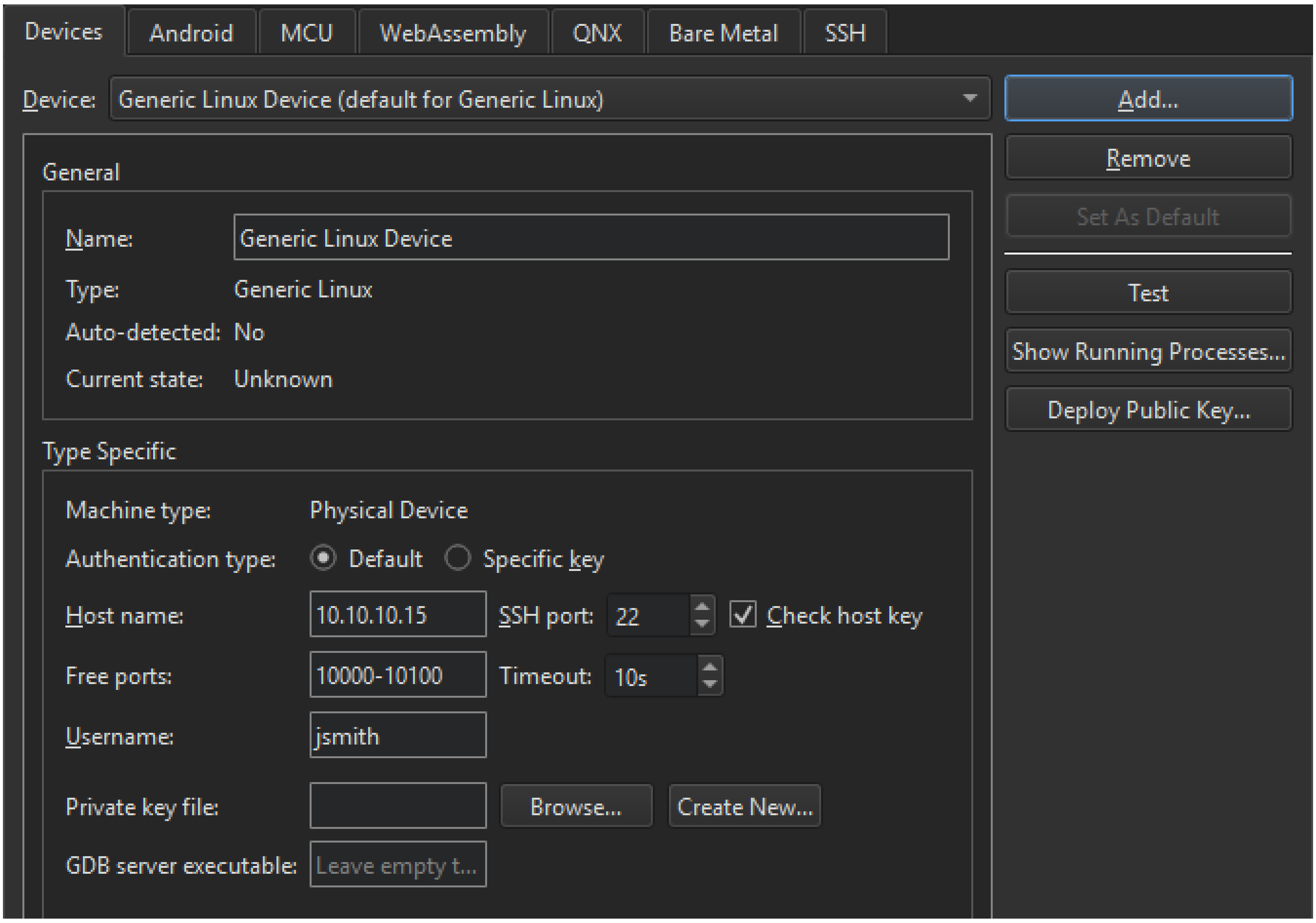
连接通用远程 Linux 设备

您可以将通用Linux设备连接到开发PC，以运行，调试和分析Qt Creator为其构建的应用程序。

如果你有一个工具链，用于在开发PC上安装嵌入式Linux设备构建应用程序，你可以将其添加到Qt Creator中。然后，您可以选择具有**嵌入式 Linux** 设备类型的工具包，以便为其构建应用程序并在设备上运行这些应用程序。

为了能够在通用远程 Linux 设备上运行和调试应用程序，您必须添加设备并在 Qt 创建器工具包中选择它们。

您可以使用向导来创建连接。您可以稍后在"**设备>>设备**"**中编辑**>**首选项**"**中的设置**。



Device: Generic Linux Device (default for Generic Linux)

General

Name: Generic Linux Device

Type: Generic Linux

Auto-detected: No

Current state: Unknown

Type Specific

Machine type: Physical Device

Authentication type: ● Default ○ Specific key

Host name: 10.10.10.15 SSH port: 22 ☑ Check host key

Free ports: 10000-10100 Timeout: 10s

Username: jsmith

Private key file:

GDB server executable: Leave empty t...

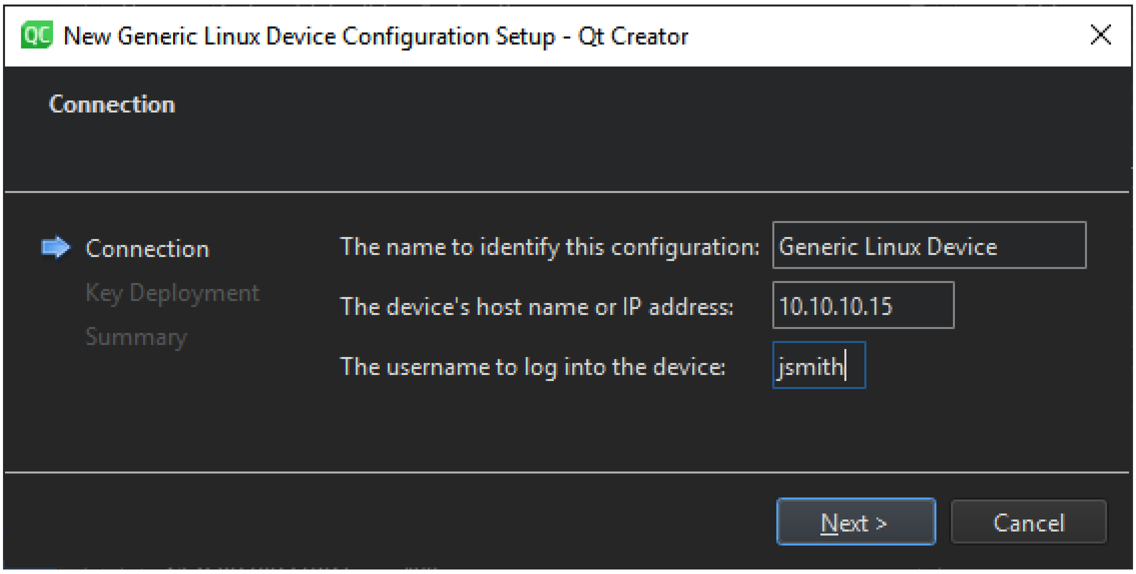
您可以使用 OpenSSH 连接来保护 Qt 创建者与设备之间的连接。开放SSH是一个连接工具，用于使用SSH协议进行远程登录。OpenSSH 套件不随 Qt 创建器一起提供，因此您必须下载它并将其安装在开发 PC 上。然后，您必须在 Qt 创建器中配置工具的路径。有关更多信息，请参阅 配置 SSH 连接。

您需要密码或 SSH 公钥和私钥对进行身份验证。如果您没有 SSH 密钥，可以使用 ssh-键健工具在 Qt 创建器中创建它。有关更多信息，请参阅 生成 SSH 密钥。

> 码，或者，如果启用了缓存，则在每次Qt Creator重新启动时输入密码。

要配置 Qt 创建器与嵌入式 Linux 设备之间的连接，并指定设备的构建和运行设置，请执行以下操作：

1. 确保可以通过 IP 地址访问您的设备。
2. 选择“**编辑**>**首选项**>**套件**> Qt **版本**”>“**添加**”以添加适用于嵌入式 Linux 的 Qt 版本。
3. 选择“**编辑**>**首选项**”>“**工具包**”>“**编译器**”>“**添加**”以添加用于构建应用程序的编译器。
4. 要部署应用程序并在设备上远程运行它们，请指定用于访问设备的参数：
   A. 选择“>**设备>设备编辑**>**首选项**”>“**添加**>**通用** Linux **设备**>**启动向导**”。

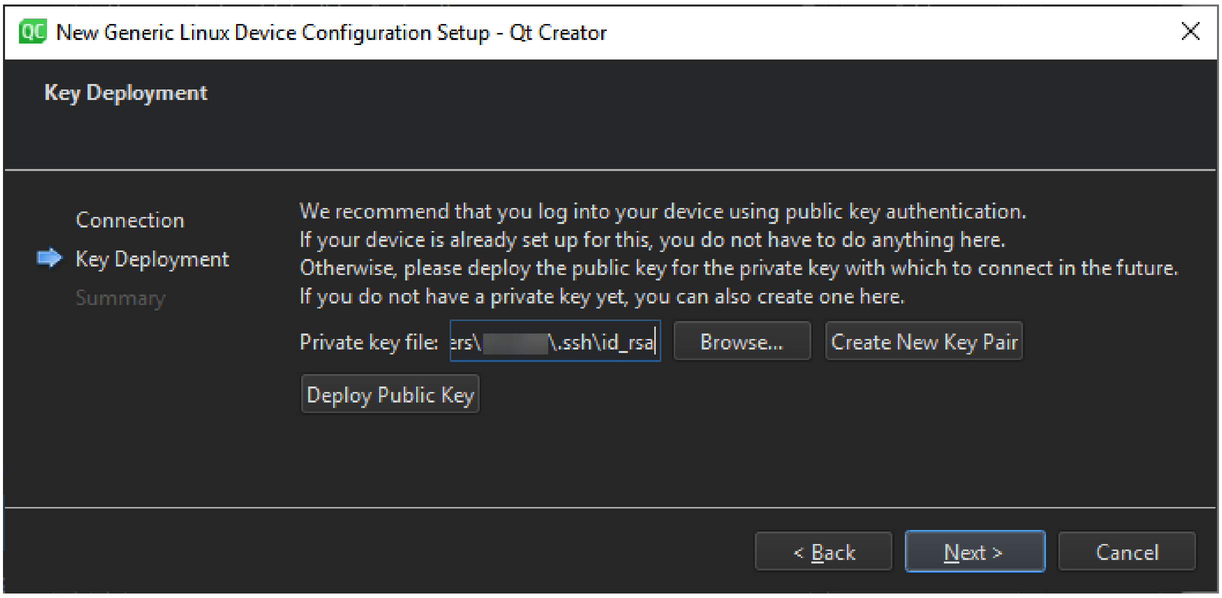


The name to identify this configuration: Generic Linux Device

The device's host name or IP address: 10.10.10.15

The username to log into the device: jsmith

   B. 在“**用于标识此配置的名称**”字段中，输入连接的名称。
   C. 在**设备的主机名或 IP 地址**字段中，输入设备的主机名或 IP 地址。此值将在变量 中可用。%{Device:HostAddress}
   D. 在“**要登录到设备的用户名**”字段中，输入用于登录到设备并以此身份运行应用程序的用户名。此值将在变量 中可用。%{Device:UserName}
   E. 选择“**下一步**”以打开“**密钥部署**”对话框。



We recommend that you log into your device using public key authentication.
If your device is already set up for this, you do not have to do anything here.
Otherwise, please deploy the public key for the private key with which to connect in the future.
If you do not have a private key yet, you can also create one here.

Private key file: ers\[redacted]\.ssh\id_rsa

   F. In **Private key file**, select a private key file to use for authentication. This value will be available in the

Generating SSH Keys.

H. Select **Deploy Public Key** to copy the public key to the device.

I. Select **Next** to create the connection.

All of these parameters can be edited later, as well as additional ones that the wizard does not show because there are sensible default values. One of these is the SSH port number, which is available in the variable .% `{Device:SshPort}`

5. Select **Edit** > **Preferences** > **Kits** > **Add** to add a kit for building for the device. Select the Qt version, compiler, and device that you added above, and choose **Generic Linux Device** for the device type.
6. To specify build settings:
   1. Open a project for an application you want to develop for the device.
   2. Select **Projects** > **Build & Run** to enable the kit that you specified above.
7. Select **Run** to specify run settings. Usually, you can use the default settings.

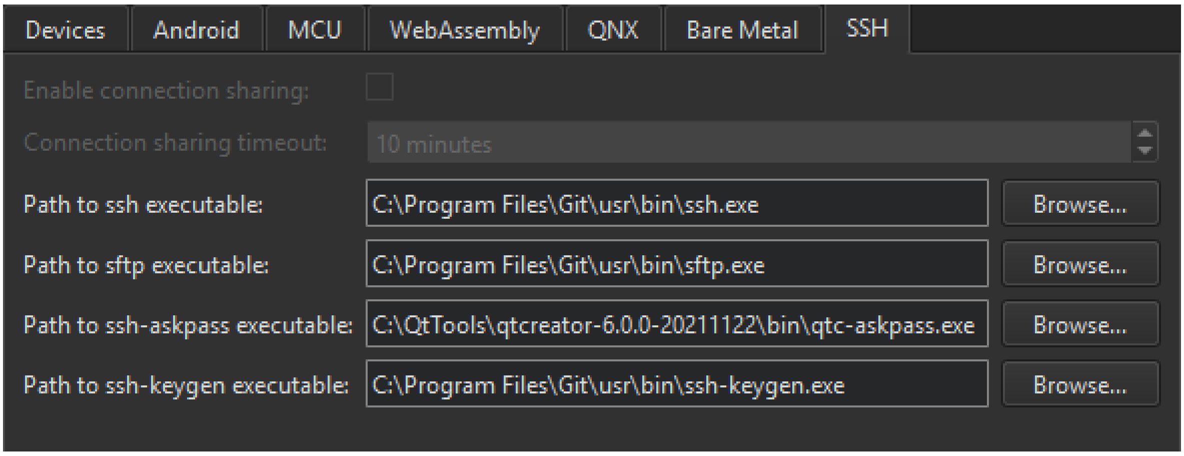
   When you run the project, Qt Creator deploys the application as specified by the deploy steps. By default, Qt Creator copies the application files to the device. For more information, see Deploying Applications to Generic Remote Linux Devices.

Configuring SSH Connections

SSH connections are established via an OpenSSH client running in master mode, if possible. Connection sharing is enabled by default to allow sharing multiple sessions over a single SSH connection. This way, a connection is only established once and then re-used by subsequent run and deploy procedures, saving connection setup overhead particularly with embedded devices. Because connection sharing is not supported on Windows, a new SSH connection is created for each deploy or run procedure.

To create SSH connections, you must install the OpenSSH suite, which includes the ssh, sftp, and ssh-keygen tools on the development PC.

To tell Qt Creator where it can find the tools, specify the paths to the directories where the tools are installed in **Edit** > **Preferences** > **Devices** > **SSH**:



Enable connection sharing: □

Connection sharing timeout: 10 minutes

Path to ssh executable: C:\Program Files\Git\usr\bin\ssh.exe

Path to sftp executable: C:\Program Files\Git\usr\bin\sftp.exe

Path to ssh-askpass executable: C:\QtTools\qtcreator-6.0.0-20211122\bin\qtc-askpass.exe

Path to ssh-keygen executable: C:\Program Files\Git\usr\bin\ssh-keygen.exe

- Deselect the **Enable connection sharing** check box to create a new SSH connection for each deploy and run procedure. This option is grayed on Windows, where connection sharing is not supported.
- In the **Connection sharing timeout** field, specify the timeout for reusing the SSH connection in minutes.
- In the **Path to ssh executable** field, enter the path to the directory where the OpenSSH executable is installed.
- In the **Path to sftp executable** field, enter the path to the directory where the SFTP executable is installed.

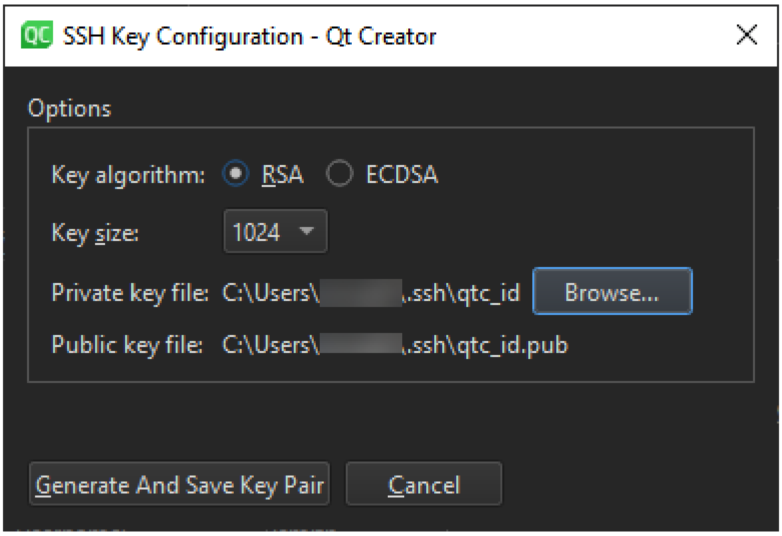
- In the **Path to ssh-keygen executable** field, enter the path to the directory where the ssh-keygen executable is installed.

Generating SSH Keys

If you do not have an SSH public and private key pair, you can generate it in Qt Creator. The connection wizard can create the key pair for you, or you can create it separately.

You can specify key length and the key algorithm, RSA or ECDSA. If you only use the keys to protect connections to the emulator or device, you can use the default values.

1. Select **Edit** > **Preferences** > **Devices** > **Devices** > **Create New**.



SSH Key Configuration - Qt Creator

Options

Key algorithm: ● RSA ○ ECDSA

Key size: 1024

Private key file: C:\Users\[redacted].ssh\qtc_id

Public key file: C:\Users\[redacted].ssh\qtc_id.pub

2. In the **Private key file** field, select the location to save the private key.

   The **Public key file** field displays the location to save the corresponding public key.

3. Select **Generate And Save Key Pair** to generate and save the keys at the specified locations.

Managing Device Processes

You can view processes running on devices and kill them. Select **Edit** > **Preferences** > **Devices** > **Devices** > **Show Running Processes**.

You can filter the processes by name or ID in the **List of Processes** dialog.

To update the process list, select **Update List**.

To kill a process, select it in the list, and then select **Kill Process**.

< Adding Docker Devices

Connecting iOS Devices >