目录

[关于openHAB 3](#_Toc494034662)

[openHAB结构 3](#_Toc494034663)

[openHAB社区 4](#_Toc494034664)

[概念 4](#_Toc494034665)

[事物，渠道，项目和链接 4](#_Toc494034666)

[事情 5](#_Toc494034667)

[事情状态 5](#_Toc494034668)

[状态转换 6](#_Toc494034669)

[状态详情 7](#_Toc494034670)

[事情状态API 9](#_Toc494034671)

[项目 10](#_Toc494034672)

[状态和命令类型格式 11](#_Toc494034673)

[DateTime 11](#_Toc494034674)

[DecimalType，PercentType 11](#_Toc494034675)

[HSBType 11](#_Toc494034676)

[PointType 11](#_Toc494034677)

[Enum Types 11](#_Toc494034678)

[关于接受多种状态数据类型的项目的说明 12](#_Toc494034679)

[事情发现 13](#_Toc494034680)

[收件箱 13](#_Toc494034681)

[音频和声音 13](#_Toc494034682)

[安装概述 15](#_Toc494034683)

[平台建议 15](#_Toc494034684)

[先决条件 15](#_Toc494034685)

[设置变体 16](#_Toc494034686)

[安装 16](#_Toc494034687)

[先决条件 16](#_Toc494034688)

[安装 17](#_Toc494034689)

[文件位置 20](#_Toc494034690)

[备用 21](#_Toc494034691)

[更新openHAB运行时 21](#_Toc494034692)

[启动openHAB作为服务 22](#_Toc494034693)

[连接到openHAB控制台 26](#_Toc494034694)

[附加步骤 28](#_Toc494034695)

关于openHAB

openHAB是将不同家庭自动化系统和技术集成到一个单一解决方案中的软件，可以实现超控自动化规则，并提供统一的用户界面。

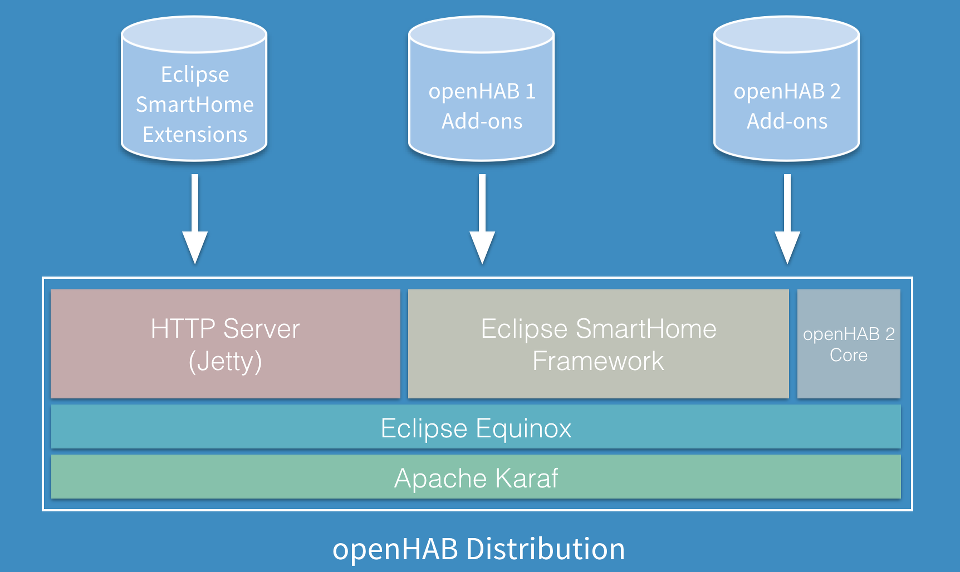
但openHAB也是用户，贡献者和维护者的**社区** 。

# openHAB结构

openHAB 2是基于[Eclipse SmartHome](https://translate.googleusercontent.com/translate_c?depth=1&hl=zh-CN&prev=search&rurl=translate.google.co.jp&sl=en&sp=nmt4&u=https://eclipse.org/smarthome/&usg=ALkJrhgZVwM3DBsQG_GZptwN_m5BBjwH_g)框架的开源解决方案。 它完全用Java编写，并将[Apache Karaf](https://translate.googleusercontent.com/translate_c?depth=1&hl=zh-CN&prev=search&rurl=translate.google.co.jp&sl=en&sp=nmt4&u=http://karaf.apache.org/&usg=ALkJrhgviv6sqgRP67OYL13Frk4kV2BsrA)与[Eclipse Equinox](https://translate.googleusercontent.com/translate_c?depth=1&hl=zh-CN&prev=search&rurl=translate.google.co.jp&sl=en&sp=nmt4&u=https://www.eclipse.org/equinox/&usg=ALkJrhi0Tnvb7zYp-WV-Azenaf21Jj8Kfw)一起用作OSGi运行时，并将其与[Jetty](https://translate.googleusercontent.com/translate_c?depth=1&hl=zh-CN&prev=search&rurl=translate.google.co.jp&sl=en&sp=nmt4&u=https://www.eclipse.org/jetty/&usg=ALkJrhh9ltiJXLNOqMhUNH5Gm_iWoKEJcA)捆绑为HTTP服务器。

openHAB是高度模块化的软件，这意味着基础安装（“运行时”）可以通过不同种类的“附件”进行扩展，以便与新的家庭自动化解决方案进行通信，或提供新的用户界面。

openHAB的结构可以这样总结：



附加组件可以来自三个不同的起源：

* + - 来自Eclipse SmartHome项目的附加组件，因为Eclipse SmartHome是openHAB 2的基础，
    - 来自openHAB版本1软件包的附加组件，因为openHAB 2包括兼容性层，
    - 为openHAB 2添加的附件。

# openHAB社区

openHAB不仅是一个解决方案，也是一个社区。

这个社区的中心是openHAB社区论坛 。

如果您有关于使用openHAB的问题，在报告问题之前，或者有关文档的意见和贡献，请查看openHAB社区论坛以获得解决方案和帮助。 更多的人会看到您的问题，当它发布在论坛，社区成员可能可以帮助您找到已知的解决方案/解决方案为您的问题。

关于发布问题和捐款准则的规定见附录 。

# 概念

Eclipse SmartHome严格区分系统的物理视图和功能视图。 虽然设置，配置，故障排除等需要物理视图，但功能方面涵盖了应用程序等重要信息，例如用户界面和自动化逻辑。

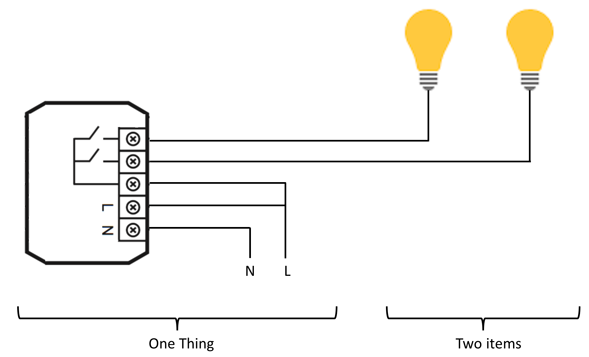
# 事物，渠道，项目和链接

事物是可以实际添加到系统的实体，并且可以一次性提供许多功能。 重要的是要注意，事情不必是设备，而是还可以表示Web服务或任何其他可管理的信息和功能来源。 事情通过一组渠道提供他们的功能。 渠道是“被动的”，可以被视为一种事物的声明，它可以提供什么。 这取决于个人设置，哪些渠道通过项目积极使用（见下文）。

项目代表应用程序使用的（细粒度）功能 - 作为用户界面或自动化逻辑。 物品有状态，可以接收命令。

物品和物品之间的粘合是链接 。 链接是正好一个Thing Channel和一个Item之间的关联。 如果通道链接到项目，则它是“启用”，这意味着项目表示的功能通过给定的通道进行处理。 渠道可以链接到多个项目和项目可以链接到多个渠道。

为了说明这些概念，请采用一个双通道执行器来控制两个灯：



执行器是Thing。这可能安装在电气柜中，它具有物理地址，需要进行设置和配置才能使用。用户对两个灯具感兴趣，这两个灯位于他家的不同位置。这些灯是所需的功能，因此项目和它们被链接到执行器的通道。甲链路可以被认为象在这个例子中的物理导线。

# 事情

事物是物理上可以添加到系统中的实体，并且可以在其中提供许多功能。重要的是要注意，事情不必是设备，而是还可以表示Web服务或任何其他可管理的信息和功能来源。从用户的角度来看，它们与设置和配置过程相关，但不适用于操作。

事情可以有配置属性，可以是可选的或强制的。这样的属性可以是诸如IP地址，Web服务的访问令牌或改变其行为的设备特定配置之类的基本信息。

事物提供了“渠道”，代表了他们提供的不同功能。通道链接到项目，其中这些链接是虚拟和物理层之间的粘合。一旦建立了这样一个链接，一个事物会对连接到其一个通道的项目发送的事件做出反应。同样地，它会积极地发送与其频道相关的项目的事件。

一种特殊的东西就是“桥”。桥梁是需要添加到系统中才能访问其他东西的东西。一个桥梁的典型例子是一些基于非IP的家庭自动化系统的IP网关。

由于可以自动发现许多事物，因此可以使用特殊的机制来处理自动发现的事物。

# 事情状态

每个东西都有一个状态对象，这有助于识别设备或服务的可能问题。下表提供了不同状态的概述：

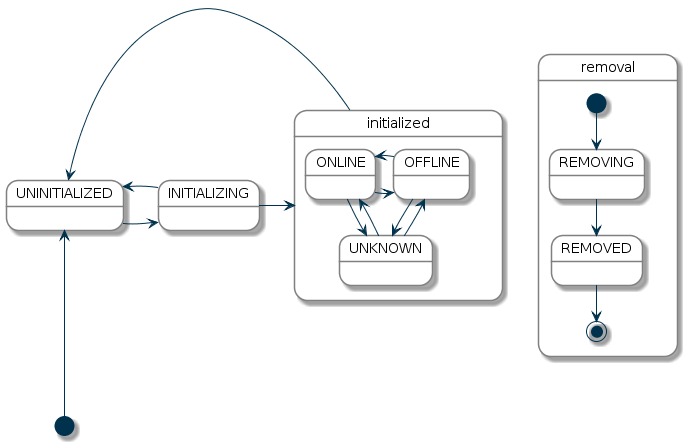
|  |  |
| --- | --- |
| 状态 | 描述 |
| UNINITIALIZED | 这是一个事情的初始状态，当它被添加或框架正在启动时。如果初始化过程失败或绑定不可用，也会分配此状态。发送到频道的命令将不被处理。 |
| INITIALIZING | 这个状态是在绑定初始化事物时分配的。这取决于初始化过程需要多长时间的绑定。发送到频道的命令将不被处理 |
| UNKNOWN | 处理程序被完全初始化，但是由于所表示的设备/服务的性质，它不能真正地告诉事件是ONLINE还是OFFLINE。因此，事情可能已经正常工作，可能会处理命令也可能不会处理命令。但是框架允许发送命令，因为如果发送命令，某些基于无线电的设备可能会在线。处理程序应该尽快将事物切换到ONLINE或OFFLINE。 |
| ONLINE | 假设某个东西所代表的设备/服务正常工作，可以处理命令。 |
| OFFLINE | 假定一件东西代表的设备/服务不能正常工作，可能不会处理命令。但是框架允许发送命令，因为一些基于射频的设备可能会返回到ONLINE，如果发送命令。 |
| REMOVING | 应该删除一个东西代表的设备/服务，但绑定没有确认删除。某些绑定需要与设备通信以将其从系统中解开。事情可能不工作，命令无法处理。 |
| REMOVED | 此状态表示在框架启动REMOVING之后，由东西代表的设备/服务已从外部系统中删除。通常这个状态是一个中间状态，因为在分配了这个状态后，事情会从数据库中删除。 |

状态UNINITIALIZED，INITIALIZING和REMOVING由框架设置，其中状态UNKNOWN，ONLINE和OFFLINE是从绑定分配的。

另外，REMOVED状态由绑定设置，表示删除过程已经完成，即事情一直处于REMOVING状态。。

# 状态转换

下图显示了允许的状态转换：



事情的初始状态是UNINITIALIZED。从UNINITIALIZED，事情进入INITIALIZING。如果初始化失败，则返回到UNINITIALIZED。如果初始化成功，绑定将事件的状态设置为UNKNOWN，ONLINE或OFFLINE，这意味着事件处理程序已完全初始化。从这个州之一，事情可以重新进入UNINITIALIZED，REMOVING或REMOVED。还可以从任何其他州到达“REMOVING”和“REMOVED”状态。

# 状态详情

状态详细信息对象进一步详细说明。下表列出了每种状态的不同状态：

|  |  |  |
| --- | --- | --- |
| UNINITIALIZED | HANDLER\_MISSING\_ERROR | 无法初始化处理程序，因为负责的绑定不可用或不启动。 |
| HANDLER\_REGISTERING\_ERROR | 处理程序在服务注册阶段失败。 |
| HANDLER\_CONFIGURATION\_PENDING | 处理程序已注册，但由于缺少配置参数而无法初始化。 |
| HANDLER\_INITIALIZING\_ERROR | 处理程序在初始化阶段失败。 |
| BRIDGE\_UNINITIALIZED | 与此事物相关联的桥梁未初始化。 |
| INITIALIZING | NONE | 没有进一步的状态详细信息。 |
| UNKNOWN | NONE | 没有进一步的状态详细信息。 |
| ONLINE | CONFIGURATION\_PENDING | 正在等待将配置信息传输到设备。某些绑定需要与设备进行通信，以确保配置被接受。 |
| NONE | 没有进一步的状态详细信息。 |
| COMMUNICATION\_ERROR | 与设备通信时出错。这也可能只是一个暂时的错误。 |
| CONFIGURATION\_ERROR | 事物配置的问题阻止与所表示的设备或服务的通信。这个问题可以通过重新配置来解决。 |
| OFFLINE | BRIDGE\_OFFLINE | 假设事情要离线，因为相应的网桥是脱机的。 |
| FIRMWARE\_UPDATING | 目前正在运行固件更新。 |
| DUTY\_CYCLE | 这个东西目前在DUTY\_CYCLE状态，这意味着它被阻止进一步的使用。 |
| REMOVING | NONE | 没有进一步的状态详细信息。 |
| REMOVED | NONE | 没有进一步的状态详细信息 |

# 事情状态API

Thing界面定义了一种方法getStatusInfo()来检索事物的当前状态。以下代码显示如何将每个事物的状态打印到控制台中：

Collection**<**Thing**>** things **=** thingRegistry**.**getAll**();**

**for** **(**Thing thing **:** things**)** **{**

ThingStatusInfo statusInfo **=** thing**.**getStatusInfo**();**

**switch** **(**statusInfo**.**getStatus**())** **{**

**case** ONLINE:

System**.**out**.**println**(**"Thing is online"**);**

**break;**

**case** OFFLINE:

System**.**out**.**println**(**"Thing is offline"**);**

**break;**

**default:**

System**.**out**.**println**(**"Thing is in state " **+** statusInfo**.**getStatus**());**

**break;**

**}**

System**.**out**.**println**(**"Thing status detail: " **+** statusInfo**.**getStatusDetail**());**

System**.**out**.**println**(**"Thing status description: " **+** statusInfo**.**getDescription**());**

**}**

# 项目

Eclipse SmartHome在物理世界（“事情”，见下文）和应用程序之间有一个严格的分离，它是围绕“项目”（也称为虚拟层）的概念构建的。

项目表示应用程序使用的功能（主要是用户界面或自动化逻辑）。物品有状态，通过事件使用。

以下项目类型目前可用（按字母顺序）：

|  |  |  |
| --- | --- | --- |
| 项目名 | 描述 | 命令类型 |
| Color | 颜色信息（RGB） | OnOff，增加减少，百分比，HSB |
| Contact | 项目存储状态例如门/窗口触点 | 开关 |
| DateTime | 存储日期和时间 |  |
| Dimmer | 携带调光器百分比值的物品 | OnOff，增加减少，百分比 |
| Group | 项目嵌套其他项目/分组收集 |  |
| Number | 以数字格式存储值 | 十进制 |
| Player | 允许控制玩家（例如音频播放器） | PlayPause，NextPrevious，RewindFastforward |
| Rollershutter | 通常用于百叶窗 | UpDown，StopMove，Percent |
| String | 存储文本 |  |
| Switch | 通常用于灯（开/关） | 开关 |

组件可以根据其成员项目导出自己的状态。

* AVG显示组中项目状态的平均值。
* OR显示组的OR，通常用于显示组中的任何项是否已设置。
* 其他聚合：AND，SUM，MIN，MAX，NAND，NOR

# 状态和命令类型格式

### DateTime

使用Java SimpleDateFormat.parse(),使用第一个匹配模式解析DateTime对象：

1. yyyy-MM-dd'T'HH:mm:ss.SSSZ
2. yyyy-MM-dd'T'HH:mm:ss.SSSX
3. yyyy-MM-dd'T'HH:mm:ssz
4. yyyy-MM-dd'T'HH:mm:ss

### DecimalType，PercentType

DecimalType并且PercentType对象使用Java的BigDecimal构造函数进行转换。PercentType值范围从0到100。

### HSBType

HSB字符串值分别由色调（0-360°），饱和度（0-100％）和值（0-100％）的三个逗号分隔值组成，例如240,100,100蓝色。

### PointType

PointType字符串由三个DecimalType以逗号分隔的字符串组成，分别表示纬度和经度（以度计）和高度（以米为单位）。

### Enum Types

|  |  |
| --- | --- |
| 类型 | 支持的值 |
| IncreaseDecreaseType | INCREASE， DECREASE |
| NextPreviousType | NEXT， PREVIOUS |
| OnOffType | ON， OFF |
| OpenClosedType | OPEN， CLOSED |
| PlayPauseType | PLAY， PAUSE |
| RewindFastforwardType | REWIND， FASTFORWARD |
| StopMoveType | STOP， MOVE |
| UpDownType | UP， DOWN |

# 关于接受多种状态数据类型的项目的说明

有许多项，其接受多个状态的数据类型，例如DimmerItem，它接受OnOffType和PercentType，RollershutterItem，它接受PercentType和UpDownType，或ColorItem，它接受HSBType，OnOffType和PercentType。由于项目具有SINGLE状态，因此可以将这些多种数据类型视为与该状态不同的视图。携带有关状态的最多信息的数据类型通常用于保留项目的内部状态，其他数据类型将从该主数据类型转换。这个主数据类型通常是返回的列表中的第一个元素Item.getAcceptedDataTypes()。

以下是上述示例的简短表格：

|  |  |  |
| --- | --- | --- |
| 项目名 | 主数据类型 | 额外的数据类型转换 |
| Color | HSBType | <ul> <li> OnOffType- OFF如果亮度HSBType等于0，ON否则</ li> <li> PercentType- HSBType</ li> </ ul>中亮度级别的值 |
| Dimmer | PercentType | OnOffType- OFF如果百分比类型指示的亮度等于0，ON否则 |
| Rollershutter | PercentType | UpDownType- UP如果由百分比类型指示的快门级别等于0，DOWN如果等于100，以及UnDefType.UNDEF任何其他值 |

# 事情发现

许多设备，技术和系统可以在网络上自动发现或通过某些API浏览。因此，智能家居解决方案使用这些功能非常有意义。

在Eclipse中，SmartHome绑定因此会为发现服务提供发现服务。所有发现结果被视为对用户的建议，并被放入收件箱。

# 收件箱

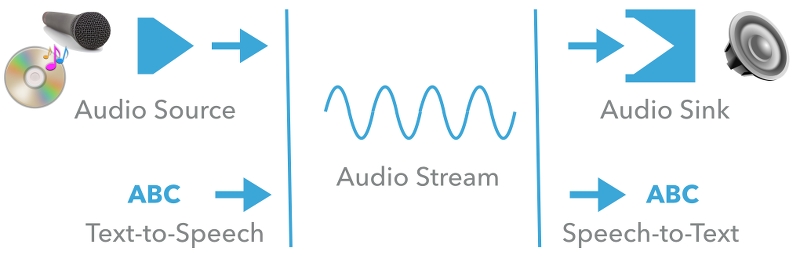
收件箱包含DiscoveryResult所有活动发现服务中所有发现的东西（）的列表。发现结果表示特定事物类型的发现的东西，可以被实例化为一个东西。结果通常包含识别发现的东西，进一步像IP地址或序列号的属性。每个发现结果在添加到收件箱中或在收件箱中更新时也会有一个时间戳，并且还可能包含一段时间，指示自动从收件箱中删除的时间。

发现结果可以被忽略或被批准，在后一种情况下，为它们创建一个事物，并且它们在应用程序中可用。如果一个条目被忽略，它将被隐藏在收件箱中，而不会为它创建一个东西。

# 音频和声音

音频和语音功能是任何智能家居解决方案的重要方面，因为它是与用户进行交互的非常自然的方式。

Eclipse SmartHome配有一个非常模块化的架构，支持各种不同的用例。它的核心是音频流的概念。音频流由音频源提供，并由音频接收器消耗。



* 音频流本质上是具有给定音频格式的字节流。它们不需要被限制在大小上，即它也被允许具有连续的流，例如来自麦克风或来自因特网电台的输入。
* 音频格式定义容器（例如WAV），编码，比特率，采样频率和深度以及位顺序（小或大端）。
* 音频源是能够产生音频流的服务。他们可以支持不同的格式，并根据请求提供请求格式的流。典型的音频源服务是麦克风。通常情况下，预计来自它们的连续流。
* 音频接收器是接受某些格式的音频流的服务。通常，这些预期播放音频流，即它们是某种扬声器或媒体设备。
* 相对于创建音频流的能力，文本到语音（TTS）服务类似于音频源。不同之处在于，它们将一个字符串作为输入，并使用给定的语音将该字符串合成到一个口头文本。TTS服务可以提供他们支持的声音和这些声音相关联的语言环境的信息。每个声音只支持一个区域。
* 语音到文本（STT）服务类似于音频接收器，但它们不是简单地播放流，而是将其转换为纯字符串。它们提供有关支持的格式和区域设置的信息。

来自STT服务的纯文本通常不是很有用，另外还有一个人类语言解释器的概念：



一个人的语言解释需要一个字符串作为输入。然后它从中导出操作（如发送命令到设备）和/或用字符串进行回复，这将打开实现会话的可能性。因为这样一个解释器不直接链接到音频流，而是仅在字符串上运行，这可以是任何类型的助手的基础，例如使用控制台，XMPP，Twitter或其他消息服务的聊天机器人。

应用程序可以动态选择要使用的服务，以便不同的汇可以用于不同的用例。默认值可以设置为所有这些服务的配置，以防应用程序不要求任何特定的服务。

# 安装概述

openHAB 2基于Eclipse SmartHome框架，完全用Java编写。因此，它只取决于可用于许多平台的Java虚拟机。openHAB可以在不同版本的Mac OS X和Windows以及许多不同版本的Linux（Ubuntu，Raspbian，...）上执行。

请注意这个事实，openHAB 2是基于一个新的核心，并引入了新的概念。因此，在互联网上可以在openHAB 1上找到的教程和帮助可能已经过时了！

# 平台建议

在本地PC或Mac上设置openHAB 2只需几步。

您获得了一些经验，并希望使用openHAB来严格控制您的家庭？典型的硬件和软件要求是：

* 全天候可用性：由以太网连接并连续运行的专用系统。
* 能源和空间高效：能够执行任务而不被夸大的装置
* 可扩展性：您的系统应该能够运行其他软件，如MQTT代理或持久性和图形化软件。
* 外设：根据您的家庭自动化硬件，您将需要其他外围设备，如WiFi接口或特殊的USB无线电模块。

许多设备都适合主持openHAB的连续安装2. 社区论坛硬件部分可以找到不同设备和环境的体验。

在树莓派的最小充分设备是相当受欢迎，尤其是作为我们提供了一个快速设置与openHABian。一个受欢迎的替代方案是我们的Synology DiskStation解决方案，许多用户已经在家里拥有。前面提到的openHabian还可以用于在现有的基于debian / ubuntu的Linux系统上启动您的openHAB 2体验。

# 先决条件

确保您的主机系统上安装了最新的Java平台。Zulu目前是推荐的用于openHAB的Java平台，尽管Oracle Java也适用于大多数配置。可以使用OpenJDK，但是它对openHAB 有一些已知的限制。

确保为ARM平台使用32位版本的JVM，即使在64位操作系统上。串行连接将不能与64位JVM一起使用，从而防止像Z-Wave这样的绑定运行。但是在X86的windos平台上可以使用64位平台。

至于现在，不推荐使用预发行Java 9安装，并且openHAB 2尚不支持安装。

为了获得最佳兼容性，即使用openHAB云服务myopenhab.org，最低推荐的Orcacle Java 8版本为“101”。注意：大多数软件包管理员都服务于较旧的版本。通过打开命令行控制台并键入java -version以下内容检查您当前的Java版本：

java version "1.8.0\_121"

Java(TM) SE Runtime Environment (build 1.8.0\_121-b13)

Java HotSpot(TM) Client VM (build 25.121-b13, mixed mode)

# 设置变体

在你开始之前，必须做出三个决定：

1. openHAB 2可作为独立于平台的存档文件或通过软件包存储库提供：

* 手动设置：下载并提取独立于平台的ZIP存档：Mac OS X，Windows，Linux，
* 软件包设置：通过软件包存储库进行安装，包括自动更新。此选项仅适用于Debian或Ubuntu衍生产品以及推荐的选择：Linux（apt / deb）

1. 稳定的释放或切削刃：

* 稳定：使用最新的官方版本（托管在Bintray）。
* 快照：受益于每日创建的快照（托管在CloudBees上）的最新更改。

# 安装

本次只记录Windos上的安装。

以下说明将指导您完成设置openHAB 2的过程。

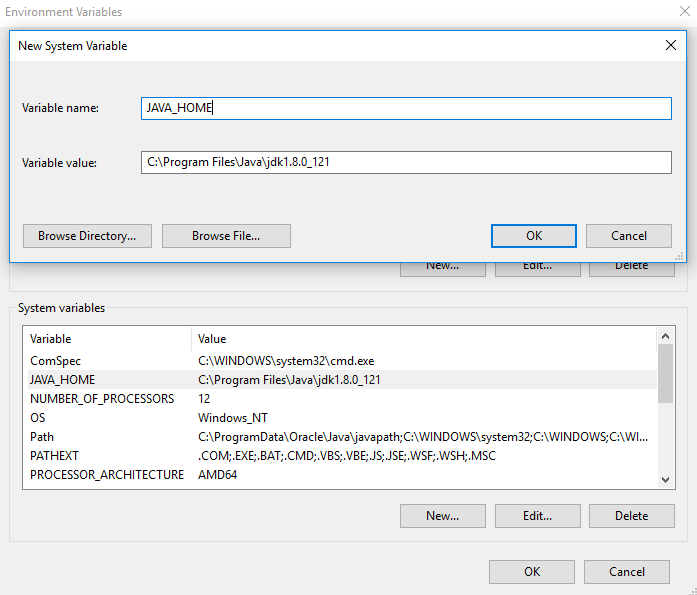
# 先决条件

首先，请确保您的系统具有必要的先决条件。

JAVA\_HOME在Windows中添加系统环境变量。导航：

控制面板➡系统和安全性➡️系统➡️高级系统设置➡️环境变量➡️系统变量

基于您的Java安装目录创建一个新的系统变量：



openHAB安装的目录必须不包含空格。在这个页面上，我们将C:\openHAB2用作一个例子。

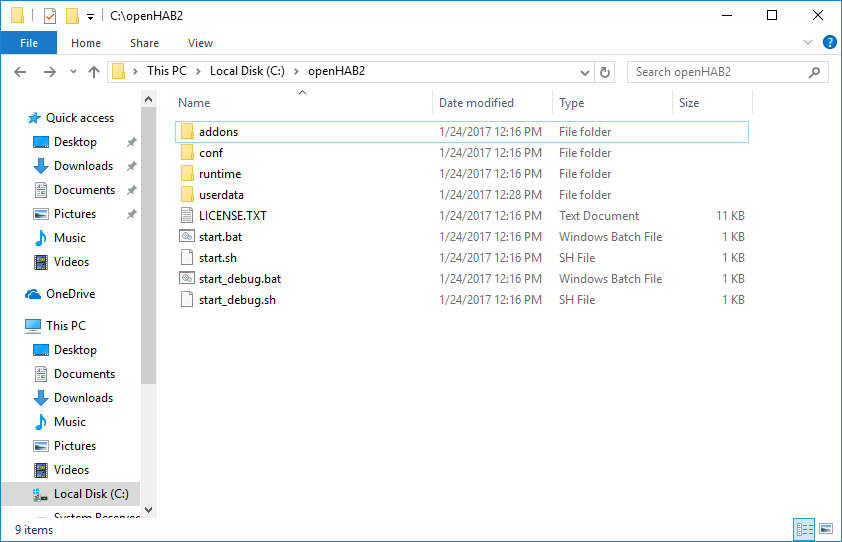
# 安装

openHAB运行时是使用平台无关的zip文件分发的。要安装它，请按照以下简单步骤操作：

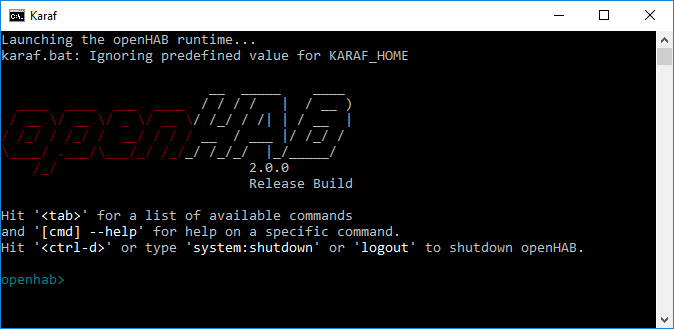
1. 选择稳定版本下载（<https://bintray.com/openhab/mvn/download_file?file_path=org%2Fopenhab%2Fdistro%2Fopenhab%2F2.1.0%2Fopenhab-2.1.0.zip>）

或下载openHAB 的最新快照版本（https://openhab.ci.cloudbees.com/job/openHAB-Distribution/lastSuccessfulBuild/artifact/distributions/openhab/target/openhab-2.2.0-SNAPSHOT.zip）

1. 将文件解压缩到您选择的目录中（例如C:\openHAB2）



1. 启动服务器：通过执行脚本启动运行时C:\openHAB2\start.bat，等待一段时间才能启动并完成。



1. 将浏览器指向http://localhost:8080。你应该看看openHAB 包选择页面。当您选择了适当的包后，此页面将包含UI选择屏幕。

**初始设置包选择**

当您第一次启动openHAB时，将要求您选择一个初始包。

有四个包可供选择，这里将详细介绍。

注意：如果您正在进行无头安装，还可以通过将package参数设置为conf/services/addons.cfg（/etc/openhab2/services/addons.cfg适用于apt安装）来选择软件包。

如果您跳过了程序包安装，系统将保持最小的安装程序，而不需要任何插件，这也不意味着没有任何UI。在这种情况下，您必须配置要安装的加载项addons.cfg。

**标准包装（推荐设置）**

这是普通用户推荐的软件包。它包含最常见的UI，剩下的用户可以决定他还需要什么。

这样安装包：

* 用于系统管理的纸张UI，包括项目访问
* 基本UI作为移动设备的新型现代Web UI
* HABPanel作为（例如壁挂式）平板电脑的仪表板UI

其他附加组件可以通过Paper UI安装，也可以直接在其中定义addons.cfg。

**专家套餐（最适合1.x用户）**

来自openHAB 1.x的所有用户都应该选择这个软件包。它包含作为核心1.x运行时的一部分，以及专门针对高级用户的工具。

这样安装包：

* 用于系统管理的纸张UI，包括项目访问
* 经典UI作为OpenHAB 1.x的知名网络UI
* 基本UI作为移动设备的新型现代Web UI
* HABPanel作为（例如壁挂式）平板电脑的仪表板UI
* HABmin作为强大的管理控制台，特别适用于Z-Wave设置
* 交互式REST API，可以轻松通过记录的UI来探索REST功能
* 所有可用的转换，因为它们曾经是核心1.x运行时的一部分

简单包（纯UI）

此软件包仅包含允许完全UI驱动的安装和配置过程的组件。这些是：

* 用于系统管理的纸张UI，配置为“简单链接”，即没有任何项目暴露给用户
* 在新的规则引擎来设置没有脚本的自动化规则
* HABPanel作为日常使用的仪表板UI
* 只有“native”openHAB 2绑定可用，1.x加载项被排除

警告：请注意，UI驱动的配置功能是新的，仍在开发中，因此许多功能将在即将推出的版本中得到增强。具体来说，规则引擎处于早期阶段，具有几个功能限制。openHAB的大多数功能仍然需要文本配置（例如，站点地图和复杂规则）。因此，这个软件包只能用于非常简单的设置，或作为演示者使用。

**演示包（样本设置）**

此软件包适用于演示目的，并可快速查看openHAB。它不仅安装了一些常见的绑定，还可以定义示例文本配置文件，这些文件也在公共OpenHAB Demo Server上使用。

这包括：

* 用于系统管理的纸张UI，以“简单”模式配置，使新添加的东西直接在控制UI上可用
* 基本UI作为移动设备的新型现代Web UI
* HABPanel作为平板电脑的仪表板界面
* 雅虎天气绑定，贝尔金WeMo，飞利浦Hue，Sonos，IPP，Astro，AVM！Fritz和NTP
* RRD4j持久化服务，用于本地存储时间序列
* MAP转换服务，因为这样被示例文件大量使用
* 示例配置文件demo.items，demo.things， demo.sitemap，demo.rules，demo.script，rrd4j.persist，de.map并en.map与地图变形使用

# 文件位置

假设安装成功，您现在将有各种文件夹C:\openHAB2：



# 备用

确保您定期备份conf和userdata文件夹，您可以压缩和解压缩这些文件夹，并从openHAB安装（甚至大多数版本）。当您有一个您很高兴的设置时，最好备份整个C:\openHAB2文件夹。故障后可随时使用。

# 更新openHAB运行时

目前没有Windows的自动更新脚本。要手动更新，请下载更新版本的openHAB分发zip文件，然后按照下列步骤操作：

1. 如果当前正在运行，请停止openHAB进程。
2. 如上所述备份openHAB。
3. 从现有安装中删除以下文件和文件夹：

* userdata\etc\all.policy
* userdata\etc\branding.properties
* userdata\etc\branding-ssh.properties
* userdata\etc\config.properties
* userdata\etc\custom.properties
* userdata\etc\distribution.info
* userdata\etc\jre.properties
* userdata\etc\org.ops4j.pax.url.mvn.cfg
* userdata\etc\profile.cfg
* userdata\etc\startup.properties
* userdata\etc\version.properties
* 任何文件userdata\etc开头org.apache.karaf
* 该userdata\cache文件夹
* 该userdata\tmp文件夹
* 该runtime文件夹

1. 复制并粘贴zip文件的内容通过现有的安装，当提示不要覆盖现有的文件

# 启动openHAB作为服务

通过在Windows中安装openHAB进程作为服务，您可以：

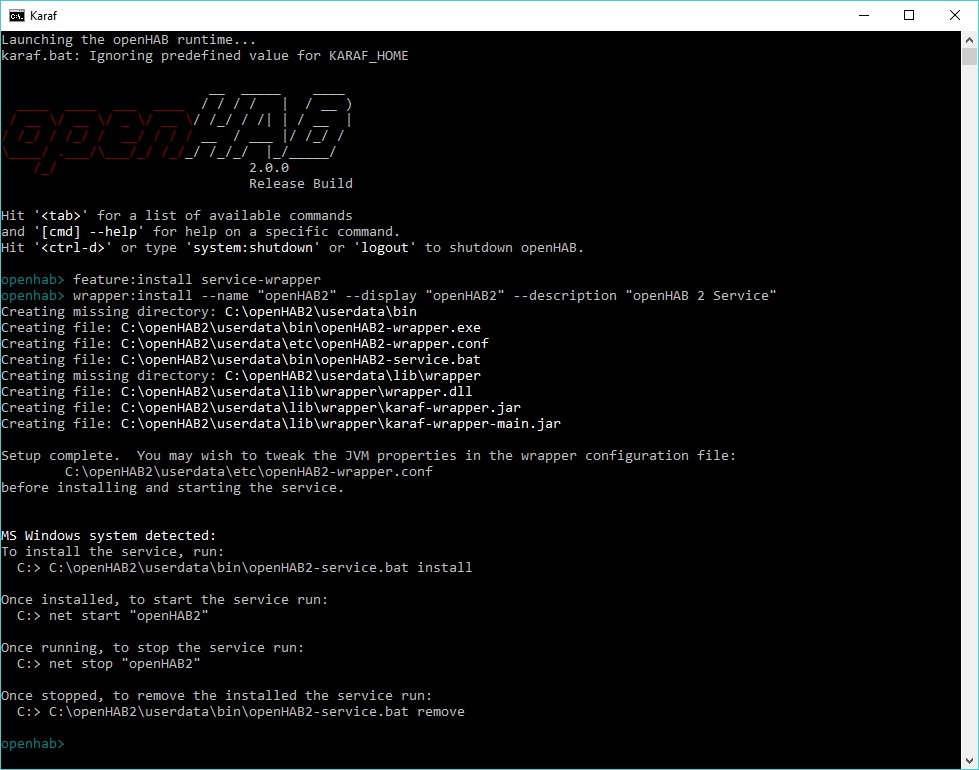
* 系统启动时自动启动
* 在后台运行

Windows服务安装步骤

1. 完成先决条件和常规安装步骤，包括软件包选择
2. 在您的openHAB控制台中发出以下两个命令：

feature:install service-wrapper

wrapper:install --name "openHAB2" --display "openHAB2" --description "openHAB 2 Service"



1. 通过键入logout当前正在运行的控制台来关闭openHAB实例。
2. 更新新创建C:\openHAB2\userdata\etc\openHAB2-wrapper.conf以包括所有必要的参数。

* 复制下面部分的所有配置文本并将其粘贴到您的openHAB2-wrapper.conf，替换所有现有的内容。
* 调整第一个条目（OPENHAB\_HOME）以匹配您的openHAB安装目录。

*#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*# openHAB2-wrapper.conf for Windows Service Installation*

*#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*# openHAB installation dir (Adapt this first setting to your system)*

set.default.OPENHAB\_HOME=C:\openHAB2

*# Wrapper Properties*

set.default.OPENHAB\_CONF=%OPENHAB\_HOME%\conf

set.default.OPENHAB\_RUNTIME=%OPENHAB\_HOME%\runtime

set.default.OPENHAB\_USERDATA=%OPENHAB\_HOME%\userdata

set.default.OPENHAB\_LOGDIR=%OPENHAB\_USERDATA%\logs

set.default.KARAF\_HOME=%OPENHAB\_RUNTIME%

set.default.KARAF\_BASE=%OPENHAB\_USERDATA%

set.default.KARAF\_DATA=%OPENHAB\_USERDATA%

set.default.KARAF\_ETC=%OPENHAB\_USERDATA%\etc

set.default.PATH=%PATH%;%KARAF\_BASE%\lib;%KARAF\_HOME%\lib

*# Java Application*

wrapper.working.dir=%KARAF\_BASE%

wrapper.java.command=%JAVA\_HOME%/bin/java

wrapper.java.mainclass=org.apache.karaf.wrapper.internal.service.Main

wrapper.java.classpath.1=%KARAF\_HOME%/lib/boot/\*.jar

wrapper.java.classpath.2=%KARAF\_DATA%/lib/wrapper/\*.jar

wrapper.java.library.path.1=%KARAF\_DATA%/lib/wrapper/

*# Java Parameters*

wrapper.java.additional.1=-Dkaraf.home="%KARAF\_HOME%"

wrapper.java.additional.2=-Dkaraf.base="%KARAF\_BASE%"

wrapper.java.additional.3=-Dkaraf.data="%KARAF\_DATA%"

wrapper.java.additional.4=-Dkaraf.etc="%KARAF\_ETC%"

wrapper.java.additional.5=-Dcom.sun.management.jmxremote

wrapper.java.additional.6=-Dkaraf.startLocalConsole=false

wrapper.java.additional.7=-Dkaraf.startRemoteShell=true

wrapper.java.additional.8=-Djava.endorsed.dirs="%JAVA\_HOME%/jre/lib/endorsed;%JAVA\_HOME%/lib/endorsed;%KARAF\_HOME%/lib/endorsed"

wrapper.java.additional.9=-Djava.ext.dirs="%JAVA\_HOME%/jre/lib/ext;%JAVA\_HOME%/lib/ext;%KARAF\_HOME%/lib/ext"

wrapper.java.additional.10=-Dopenhab.home="%OPENHAB\_HOME%"

wrapper.java.additional.11=-Dopenhab.conf="%OPENHAB\_HOME%\conf"

wrapper.java.additional.12=-Dopenhab.runtime="%OPENHAB\_HOME%\runtime"

wrapper.java.additional.13=-Dopenhab.userdata="%OPENHAB\_HOME%\userdata"

wrapper.java.additional.14=-Dopenhab.logdir="%OPENHAB\_USERDATA%\logs"

wrapper.java.additional.15=-Dfelix.cm.dir="%OPENHAB\_HOME%\userdata\config"

wrapper.java.additional.16=-Dorg.osgi.service.http.port=8080

wrapper.java.additional.17=-Dorg.osgi.service.http.port.secure=8443

wrapper.java.maxmemory=512

*# Wrapper Logging Properties*

wrapper.console.format=PM

wrapper.console.loglevel=INFO

wrapper.logfile=%OPENHAB\_USERDATA%\logs\wrapper.log

wrapper.logfile.format=LPTM

wrapper.logfile.loglevel=INFO

wrapper.logfile.maxsize=10m

wrapper.logfile.maxfiles=5

wrapper.syslog.loglevel=NONE

*# Wrapper Windows Properties*

wrapper.console.title=openHAB2

wrapper.ntservice.name=openHAB2

wrapper.ntservice.displayname=openHAB2

wrapper.ntservice.description=openHAB 2 Service

wrapper.ntservice.dependency.1=

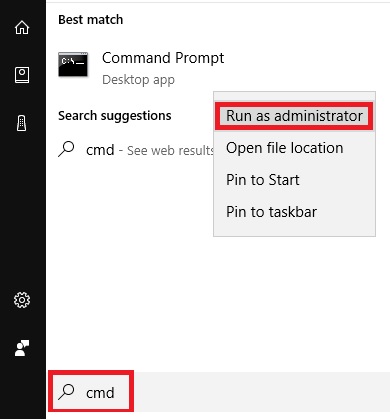
wrapper.ntservice.starttype=AUTO\_START

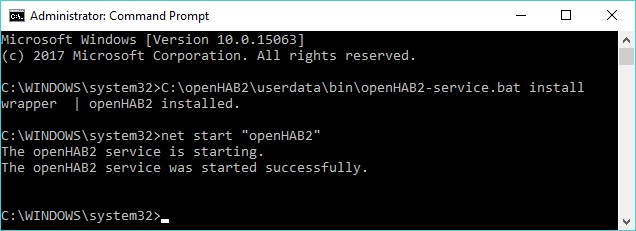
wrapper.ntservice.interactive=false

1. 打开提升的命令提示符并键入以下命令：

C:\openHAB2\userdata\bin\openHAB2-service.bat install

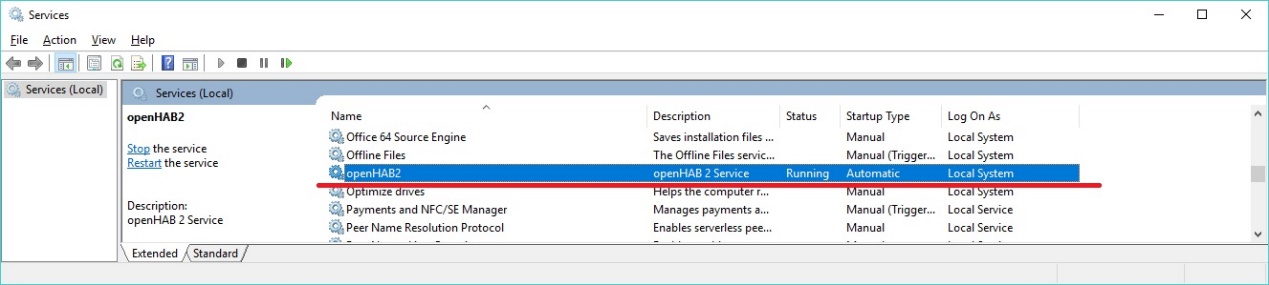
net start "openHAB2"





1. 您的openHAB Windows服务现已安装并运行。通过以下方式验证正确的操作：

* 浏览http：// localhost：8080
* 验证Windows服务是否正在运行并设置为自动启动类型。使用services.msc和查找openHAB2服务。

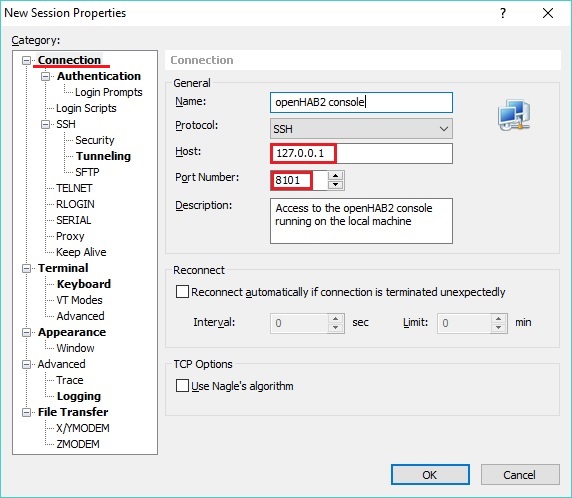


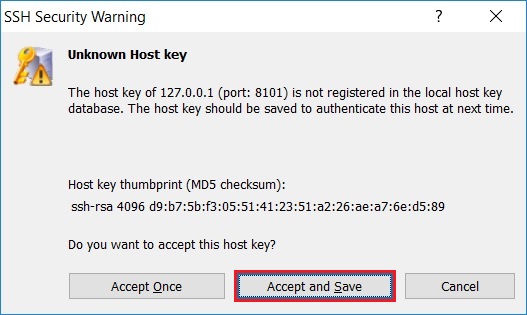
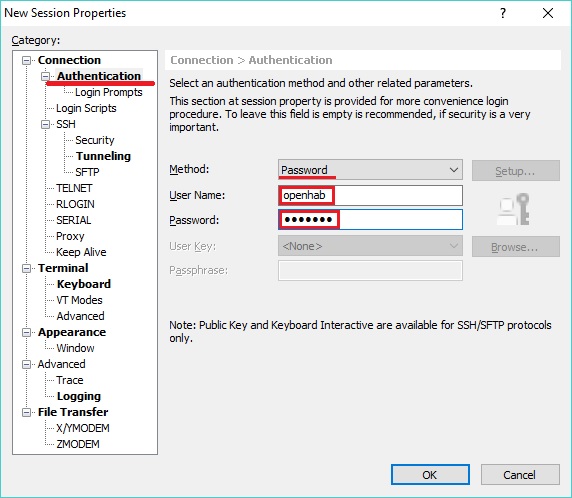
* 使用SSH客户端登录到控制台（参见下面的信息）

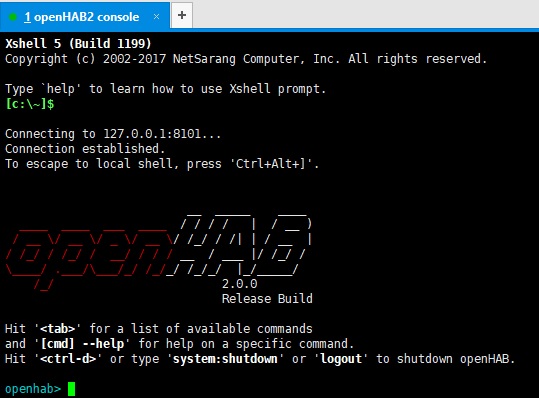
# 连接到openHAB控制台

1. 安装SSH客户端应用程序，例如Putty，KiTTY或Xshell 5
2. 使用以下参数设置会话：

* 主机：127.0.0.1
* 港口：8101
* 用户名： openhab
* 密码： habopen







# 附加步骤

OpenHAB 2设置并运行后，还有一些额外的安装步骤，您应该考虑：

* 在您的openHab主机设备上配置网络共享并在本地安装它：Linux Samba Share，Windows文件共享...
* 在本地计算机上安装Eclipse SmartHome Designer（http://docs.openhab.org/installation/designer.html），以管理（远程）配置文件。设计器内置了对openHAB语法和元素的支持。