|  |  |
| --- | --- |
|  | WebExpress  Anwendungshandbuch für Rasperry PI  Version 1.4.4.0 |

Inhalt

1 Allgemein 3

2 Lizenz 3

3 Installation 3

3.1 Betriebssystem installieren 3

3.2 Betriebssystem einrichten 4

3.3 Anwendungsprogramme einrichten 4

3.4 Statische IP festlegen 6

3.5 WebExpress installieren 7

4 WebExpress einrichten 7

4.1 WebExpress-Anwendungen installieren 7

4.2 Https einrichten 8

4.2.1 Certificate Authority (CA) erstellen 8

4.2.2 Zertifikat ausstellen 9

4.2.3 Zertifikate in WebExpress installieren 10

4.3 Grundkonfiguration 10

5 WebExpress starten 11

6 Installattion Multicast Domain Name Service (mDNS) 11

7 Zertifikate in Windows installieren 11

8 Update 13

9 Einkaufsliste 13

10 Quellen 13

# Allgemein

WebExpress ist ein leichtgewichtiger Webserver, welcher für den Einsatz in leistungsarmen Umgebungen optimiert wurde. Schon auf kleinen Systemen, wie zum Beispiel der Raspberry PI, können Webanwendungen effizient betrieben werden. Dies wird durch einen kleinen Fußabdruck mit geringer Ressourcenbelastung erreicht. Weiterhin verfügt WebExpress über ein leistungsstarkes und optimiertes Plugin-System, mit einer umfassenden API und Anwendungsvorlagen. Damit lassen sich Webanwendungen einfach und schnell in einer .Net-Sprache (z.B. C#) entwickeln. WebExpress basiert auf Kestrel, ein plattformübergreifender Webserver für ASP.NET Core. Damit unterstützt WebExpress ebenfalls:

* Https
* HTTP/2 (derzeit nicht macOS)

# Lizenz

Die Software steht als Open-Source (MIT) zur freien Verfügung. Die Softwarequellen können über <https://github.com/ReneSchwarzer/WebExpress> bezogen werden.

WebExpress baut auf Komponenten auf, welche als Open-Source zur Verfügung stehen:

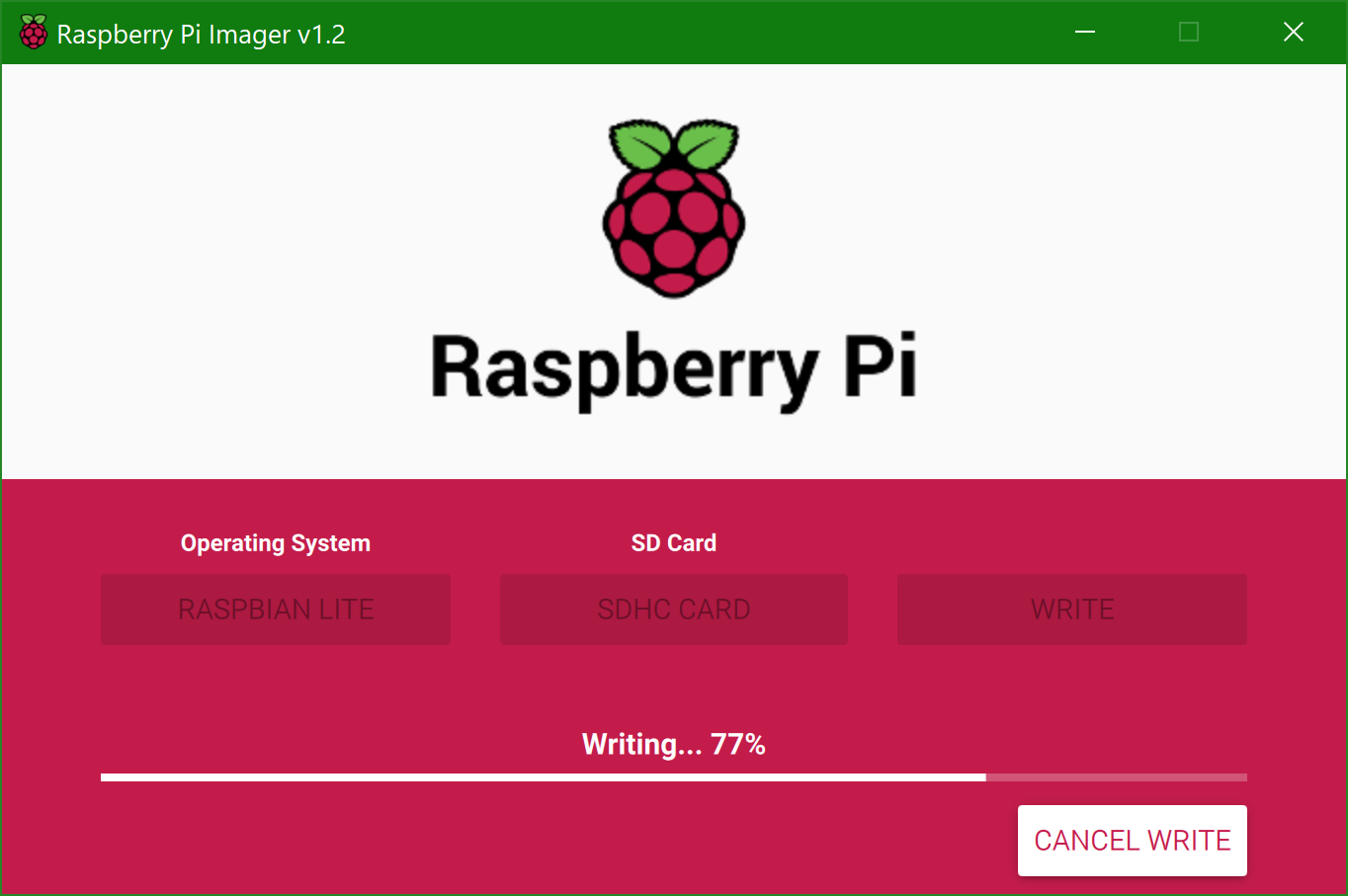
* <https://github.com/dotnet/core> (MIT)
* <https://github.com/dotnet/aspnetcore> (MIT)
* <https://getbootstrap.com/> (MIT)
* <https://www.chartjs.org> (MIT)
* <https://jquery.com/> (MIT)
* <https://summernote.org/> (MIT)
* <https://popper.js.org/> (MIT)
* <https://github.com/kurtobando/simple-tags> (MIT)
* <https://github.com/nathancahill/split/tree/master/packages/split-grid> (MIT)
* <https://github.com/uxsolutions/bootstrap-datepicker> (Apache 2.0)

# Installation

Die Installation wird anhand des Raspberry PI beschrieben. Die allgemeine Vorgehensweise kann jedoch auch auf andere Betriebssysteme angewendet werden.

## Betriebssystem installieren

Der Erste Schritt besteht darin, dass Betriebssystem auf eine SD-Karte zu schreiben. Hierzu gibt es unter <https://downloads.raspberrypi.org/imager/imager.exe> ein kostenloses Programm (Windows), mit deren Hilfe das Image auf die SD-Karte kopiert wird.



## Betriebssystem einrichten

Im zweiten Schritt wird die SD-Karte in dem Raspberry Pi eingesetzt und der Raspberry Pi gestartet. Da SSH noch nicht aktiv ist, muss eine Tastatur und ein Bildschirm angeschlossen werden. Wenn der Raspberry Pi gebootet wurde, kann das Anmelden mit

User: pi

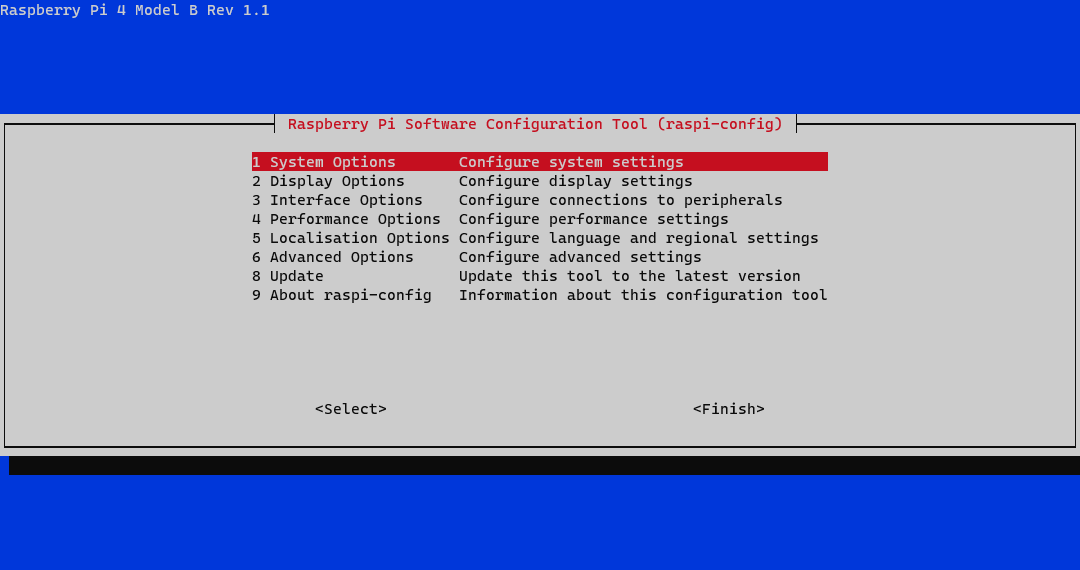
Passwort: raspberry

erfolgen.

Nach der erfolgreichen Anmeldung wird das Dienstprogramm raspi-config aufgerufen, mit deren Hilfe die Grundkonfiguration des Raspberry Pi erfolgt.

pi@raspberrypi:~ $ sudo raspi-config

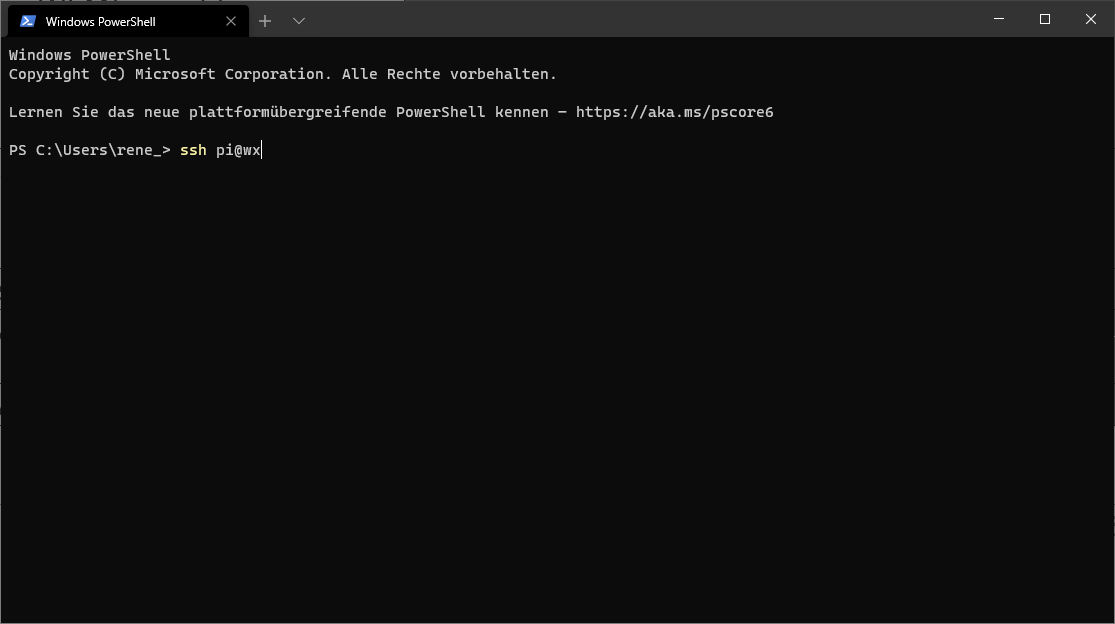
Es wird empfohlen das Passwort zu ändern, sowie bei Bedatf WLAN-einzurichten, die Zeitzone und den Hostnamen zu ändern. Im weiteren Verlauf des Anwendungshandbuches wird der Hostname wx verwendet. Darüber hinaus ist SSH zu aktivieren (zu finden unter Interface Options).



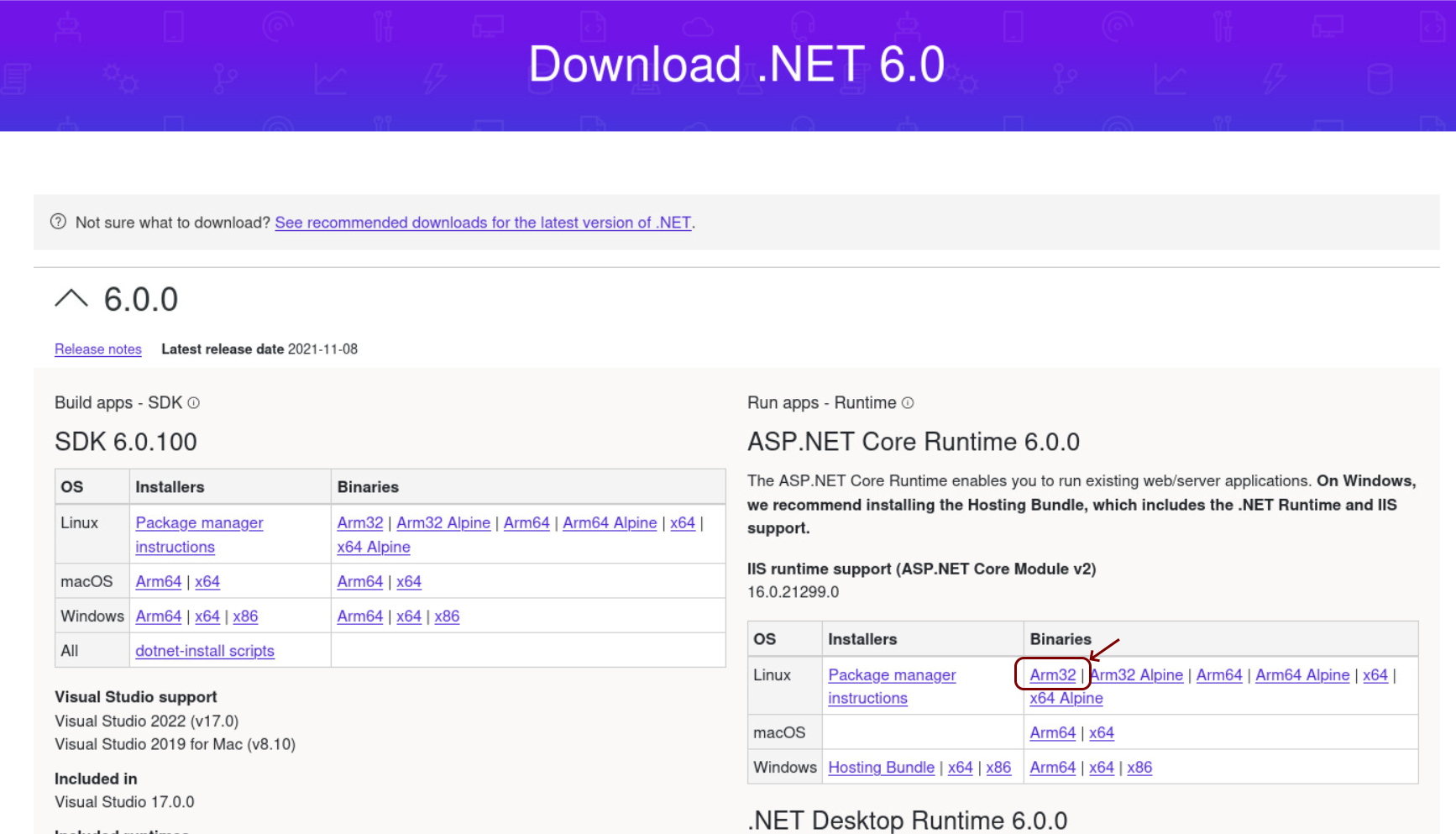
Alle nachfolgenden Schritte können nun über SSH erfolgen und der Raspberry Pi von der Tastatur und Bildschirm getrennt werden.

## .NET Runtime

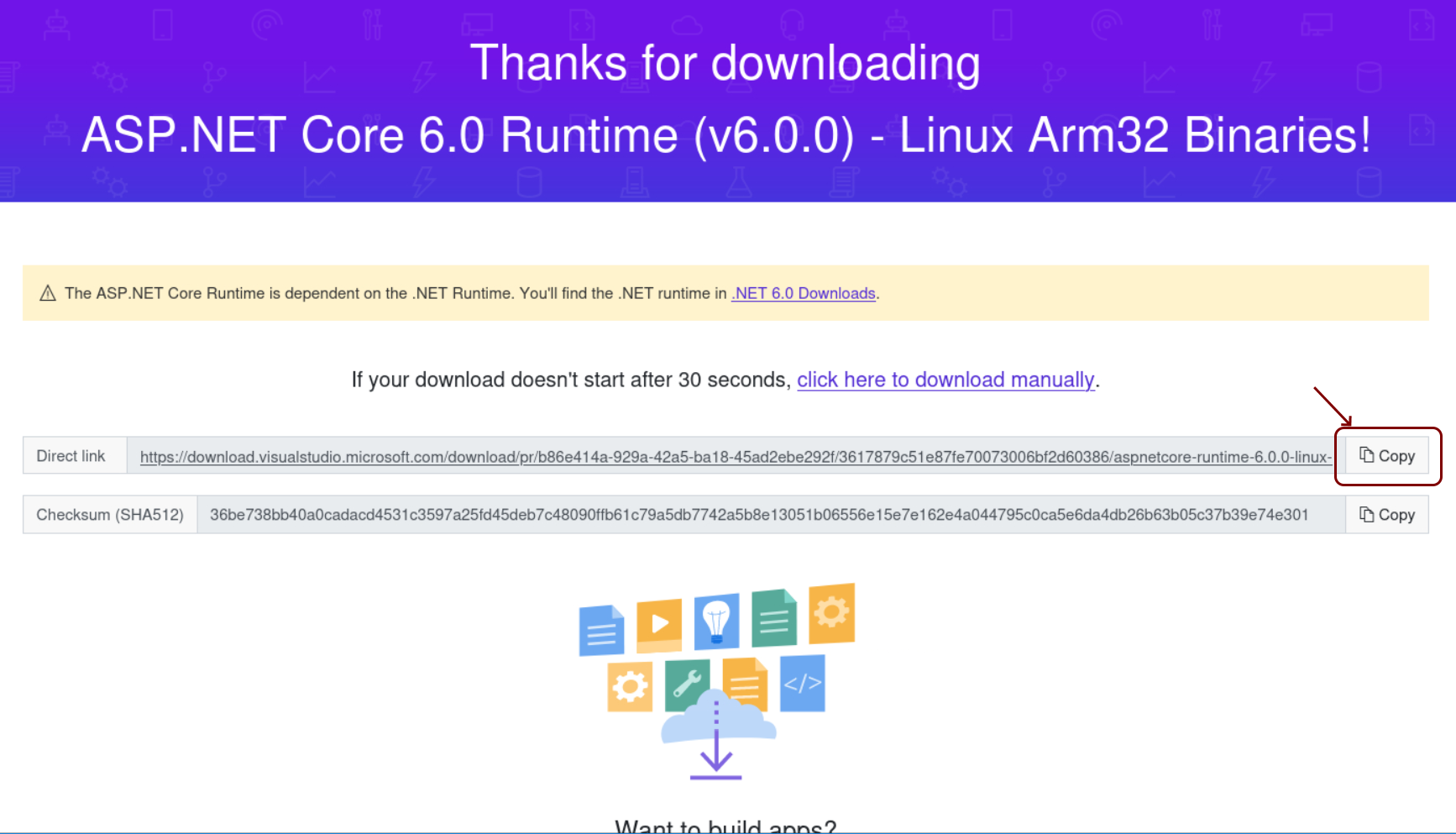
Nachdem SSH aktiviert wurde, kann mit Hilfe eines SSH-Clients (z.B. Putty, OpenSSH) eine Verbindung zum Raspberry Pi aufgebaut werden.



Zunächst muss die .NET Runtime und die ASP.NET Core Runtime installiert werden. Hilfe hierzu wird unter [1] angeboten. Die aktuellen Versionen (derzeit 6.0.100) kann von Microsoft unter <https://dotnet.microsoft.com/download/dotnet-core> kostenlos bezogen werden.



Für den Raspberry Pi sind die Binaries für Linux-Arm32 zu verwenden. Der Direkt-Link zu den Linux-Arm32 Binaries ist zu kopieren.



Auf den Raspberry werden mittels wget die Linux-Arm32-Archive, sowohl für die NET - und die ASP.NET Core Runtime, heruntergeladen.

pi@wx:~ $ wget <https://download.visualstudio.microsoft.com/download/pr/dcd635a9-6caf-4958-b3df-01e1c204ca1e/b201804d05de9573a8b454bccac56482/dotnet-runtime-6.0.0-linux-arm.tar.gz>

pi@wx:~ $ wget <https://download.visualstudio.microsoft.com/download/pr/b86e414a-929a-42a5-ba18-45ad2ebe292f/3617879c51e87fe70073006bf2d60386/aspnetcore-runtime-6.0.0-linux-arm.tar.gz>

Vorbereitend für die Installation von .NET Core muss ein Verzeichnis unter /usr/share/dotnet-sdk angelegt werden, indem die .NET Archive anschließend entpackt werden.

pi@wx:~ $ sudo mkdir /usr/share/dotnet-sdk

Nach Anlegen des Verzeichnisses /usr/share/dotnet-sdk können die Binaries entpackt werden.

pi@wx:~ $ sudo tar zxf dotnet-runtime-6.0.0-linux-arm.tar.gz -C /usr/share/dotnet-sdk/

pi@wx:~ $ sudo tar zxf aspnetcore-runtime-6.0.0-linux-arm.tar.gz -C /usr/share/dotnet-sdk/

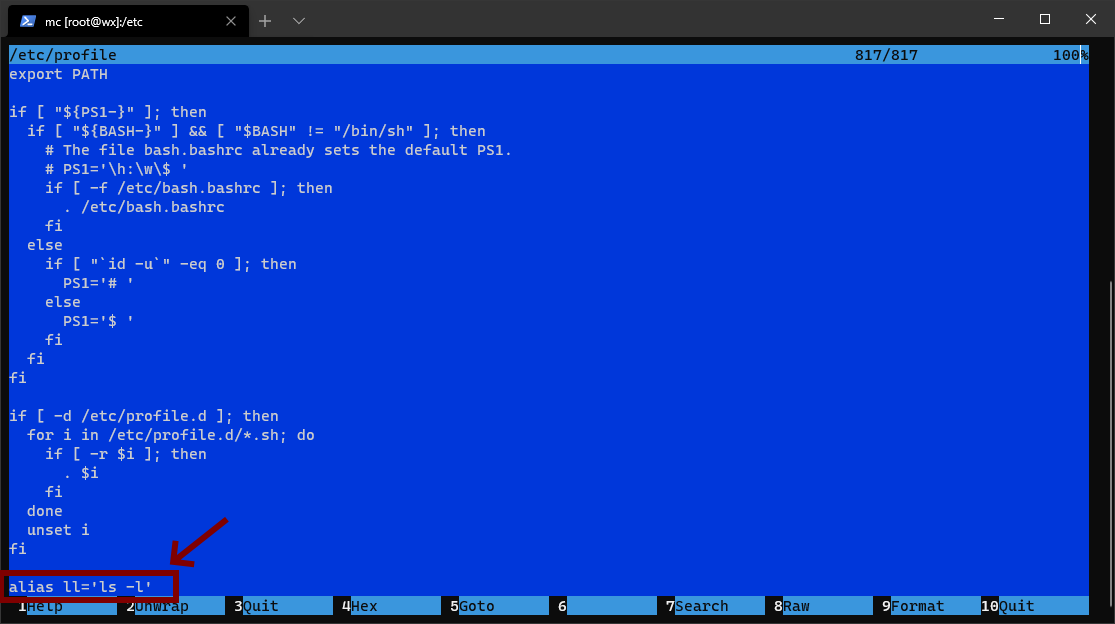
## Dienstprogramme

Im folgenden Schritt werden weitere (Dienst-)Programme installiert, welche für die Ausführung von WebExpress oder für die Administration des Raspberry Pi hilfreich sind.

Als optionale Anwendung kann der Midnight Commander (MC) installiert und das Profil angepasst werden.

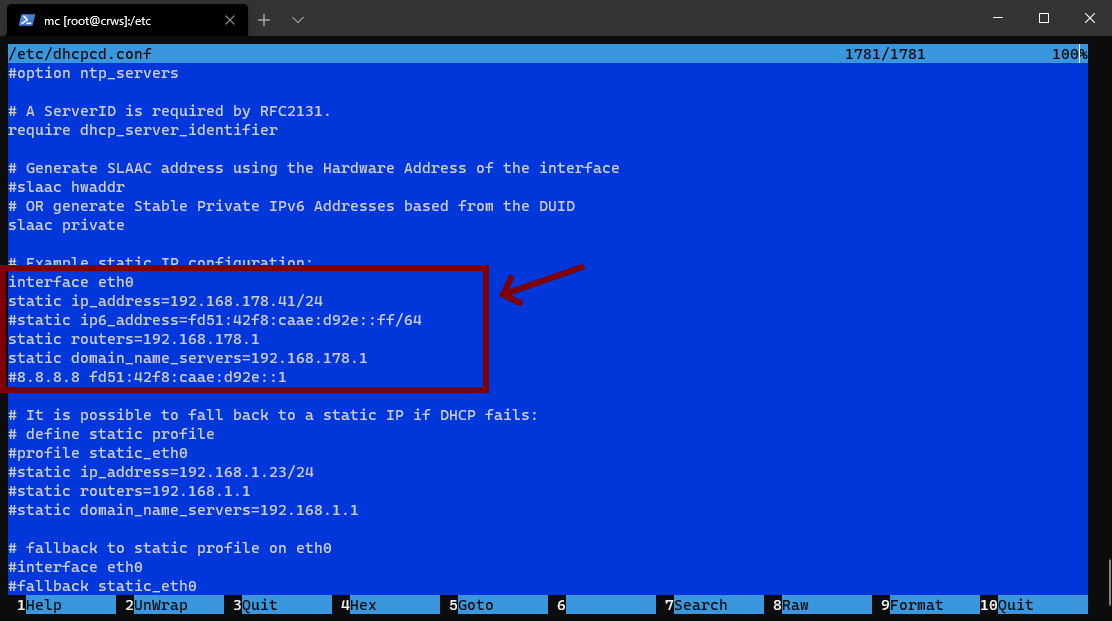
pi@wx:~ $ sudo apt-get install mc -y

Das Profil ist bei Bedarf um alias ll='ls -l' zu erweitern.



## Statische IP festlegen

Es wird empfohlen eine Statische IP-Adresse für den Raspberry unter /etc/dhcpcd.conf zu konfigurieren (siehe [2]).



## Multicast Domain Name Service (mDNS)

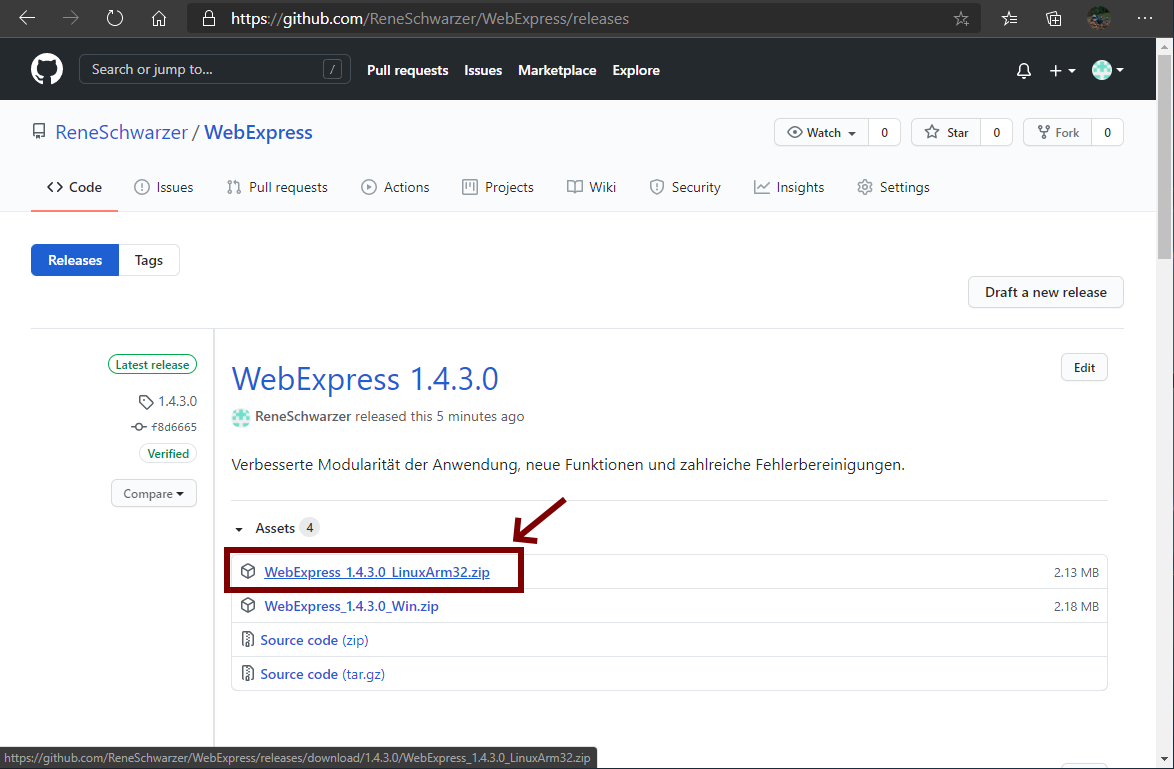
Avahi eine mDNS Open-Source-Implementierung. Geben Sie in der Eingabeaufforderung den folgenden Befehl ein, um Avahi zu installieren:

pi@wx:~ $ sudo apt install avahi-daemon -y

Sobald der Installationsvorgang abgeschlossen ist, werden lokale Netzwerkabfragen unter wx.local angenommen und beantwortet.

## WebExpress

WebExpress wird in gepackter Form für den Raspberry Pi im GitHub-Repository https://github.com/ReneSchwarzer/WebExpress/releases kostenlos bereitgestellt.



Die Binaries von WebExpress können mittels wget von GitHub bezogen werden.

pi@wx:~ $ wget https://github.com/ReneSchwarzer/WebExpress/releases/download/1.4.4.0/WebExpress\_1.4.4.0\_LiLinuxA32.zip

Als Vorbereitung für die Installation von WebExpress ist ein Verzeichnis unter /opt/wx anzulegen, indem die Binaries entpackt werden.

pi@wx:~ $ sudo mkdir /opt/wx

Anschließend ist das Archiv zu entpacken.

pi@wx:~ $ sudo unzip WebExpress\_1.4.4.0\_LinuxArm32.zip -d /opt/wx

Nachdem WebExpress erfolgreich entpackt wurde, müssen die Ausführungsrechte erteilt werden.

pi@wx:~ $ sudo chmod +x /opt/wx/webexpress.sh /opt/wx/WebExpress.App

Zum automatischen Starten der WebExpress-Anwendung ist die mitgelieferte SystemCtl-Unit zu installieren.

pi@wx:~ $ sudo cp /opt/wx/webexpress.service /etc/systemd/system

Zum Abschluss muss die SystemCtl-Unit aktiviert werden.

pi@wx:~ $ sudo systemctl enable webexpress.service

# WebExpress einrichten

Bevor WebExpress gestartet werden kann, ist dieser zu konfigurieren. Weiterhin sind WebAnwendungen zu installieren.

## Grundkonfiguration

In der Konfigurationsdatei /opt/wx/config/webexpress.config.xml werden die allgemeinen Einstellungen von WebExpress abgelegt.

|  |  |  |
| --- | --- | --- |
| Eigenschaft | Beschreibung | Beispiel |
| Endpoint | Erstellt einen Endpunkt, auf dem WebExpress lauscht und eingehende Verbindungen verarbeitet. Es können beliebig viele Endpunkte konfiguriert werden.   * **uri:** Die Uri mit dem Schema, einen Hostnamen und einen Port. Der Hostname \* steht für alle verfügbaren Endpunkte. Wird kein Port angegeben, so wird der Standardport verwendet (z.B. 443 für Https). * **pfx:** Der Keystore in Form einer pfx-Datei * **password:** Das Passwort der pfx-Datei. | <endpoint uri=http://\*/ />  <endpoint uri="https://\*:443/" pfx="./Cert/wx.pfx" password="hallo" /> |
| Limit | Legt Limitierungen des WebServers fest.   * **connectionlimit:** Anzahl der gleichzeitig aktiven Verbindungen. * **Uploadlimit:** Anzahl der Bytes, welche maximal im Body an den Webserver übertragen werden dürfen. | <limit>  <connectionlimit>  300  </connectionlimit>  <uploadlimit>  30000000  </uploadlimit>  </limit> |
| Kultur | Legt die Sprache, den verwendeten Kalender und die Formatierung für Datumsangaben und Zahlen für Ausgaben fest. | <culture>de-DE</culture> |
| Assets | Verzeichnis indem sich statische Dateien befinden, die vom WebServer ausgeliefert werden sollen. | <assets>./</assets> |
| Kontextpfad | Der Kontextpfad ist der Präfix-Pfad einer Uri (z.B. http://localhost/contextpath/pathToResource). | <contextpath>wx</contextpath> |
| Pluginverzeichnis | Verzeichnis, indem die Plugins ausgeführt werden. | <stage>./</stage> |

## Https einrichten

Eine sichere und vertrauliche Kommunikation zwischen dem WebClient und den Webserver kann durch Nutzung von Zertifikaten gewährleistet werden. Im einfachsten Fall können diese Zertifikate selbst ausgestellt werden und diese auf dem Raspberry Pi installiert werden. Weiteren Informationen sind unter [3] zu finden.

### Certificate Authority (CA) erstellen

Zum Ausstellen von Zertifikaten muss zunächst eine Certificate Authority (CA)-Stelle eingerichtet werden.

#### Privater Schlüssel

Als erstes muss ein geheimer privater Schlüssel der CA erstellt werden. Dieser wird den Namen caKey.pem erhalten.

pi@wx:~ $ openssl genrsa -out caKey.pem 4096

#### Root-Zertifikat erstellen

Im zweiten Schritt ist das Root-Zertifikat zu erstellen, welches den Namen ca.pem erhält.

pi@wx:~ $ openssl req -x509 -new -nodes -extensions v3\_ca -key caKey.pem -days 36500 -out ca.pem -sha512

Während des Erstellungsprozesses werden verschiedene Daten abgefragt und im Root-Zertifikat gespeichert.

|  |  |  |
| --- | --- | --- |
| Attribut | Bemerkung | Beispiel |
| Country Name | Der Ländercode im zweistelligen ISO-Format | DE |
| State or Province Name | Geben Sie hier Ihr Bundesland an. | Berlin |
| Locality Name | Geben Sie hier Ihre Stadt an.  Dieses Feld gibt die Stadt an, in der sich die Organisation befindet. Verwenden Sie keine Abkürzungen. Schreiben Sie beispielsweise "Saint Louis" anstelle von "St. Louis". Das Feld muss den Namen der Stadt enthalten, in der es registriert ist. | Berlin |
| Organization Name | Geben Sie hier Ihren Namen bzw. Firmennamen an.  Der Name der Organisation (Gesellschaft, Kommanditgesellschaft, Universität oder Regierungsbehörde) muss bei einer Behörde auf nationaler, staatlicher oder städtischer Ebene registriert sein. | WebExpress |
| Organizational Unit Name | Geben Sie hier Ihre Abteilung (falls vorhanden) an. |  |
| Common Name | Geben Sie hier den genauen Domainnamen an, welcher durch das Zertifikat geschützt werden soll.  Der allgemeine Name, auch als URL bezeichnet, ist der vollständig qualifizierte Domänenname, der für die DNS-Suche Ihres Servers verwendet wird (z. B. www.mydomain.com). Browser verwenden diese Informationen, um Ihre Website zu identifizieren.  Hinweis: Sie können keine Sonderzeichen (?, $,% usw.), IP-Adressen, Portnummern oder "http: // oder https: //" in Ihrem allgemeinen Namen verwenden. | WebExpress CA |
| Email Adress | Geben sie hier die E-Mail-Adresse des Verantwortlichen an. | [ca@example.com](mailto:ca@example.com) |

Ist dieser Schritt abgeschlossen, so ist die CA fertig und einsatzbereit.

## Zertifikat ausstellen

Nachdem die CA erstellt wurde, können mit deren Hilfe Zertifikate ausgestellt werden.

#### Privater Schlüssel

Für jedes Zertifikat ist ein neuer geheimer privater Schlüssel anzulegen. Für WebExpress wird ein geheimer privater Schlüssel mit den Namen wxKey.pem erstellt.

pi@wx:~ $ openssl genrsa -out wxKey.pem 4096

#### Zertifikatsanfrage erstellen

Zum Ausstellen eines Zertifikates muss eine Zertifikatsanfrage an die CA gestellt werden. Die Anfrage wird unter den Namen wx.csr gespeichert.

pi@wx:~ $ openssl req -new -key wxKey.pem -out wx.csr -sha512

Im Erstellungsprozess werde verschiedene Daten abgefragt und in der Anfrage gespeichert.

|  |  |  |
| --- | --- | --- |
| Attribut | Bemerkung | Beispiel |
| Country Name | Der Ländercode im zweistelligen ISO-Format | DE |
| State or Province Name | Geben Sie hier Ihr Bundesland an. | Berlin |
| Locality Name | Geben Sie hier Ihre Stadt an.  Dieses Feld gibt die Stadt an, in der sich die Organisation befindet. Verwenden Sie keine Abkürzungen. Schreiben Sie beispielsweise "Saint Louis" anstelle von "St. Louis". Das Feld muss den Namen der Stadt enthalten, in der es registriert ist. | Berlin |
| Organization Name | Geben Sie hier Ihren Namen bzw. Firmennamen an.  Der Name der Organisation (Gesellschaft, Kommanditgesellschaft, Universität oder Regierungsbehörde) muss bei einer Behörde auf nationaler, staatlicher oder städtischer Ebene registriert sein. | WebExpress |
| Organizational Unit Name | Geben Sie hier Ihre Abteilung (falls vorhanden) an. |  |
| Common Name | Geben Sie hier den genauen Domainnamen an, welcher durch das Zertifikat geschützt werden soll.  Der allgemeine Name, auch als URL bezeichnet, ist der vollständig qualifizierte Domänenname, der für die DNS-Suche Ihres Servers verwendet wird (z. B. www.mydomain.com). Browser verwenden diese Informationen, um Ihre Website zu identifizieren.  Hinweis: Sie können keine Sonderzeichen (?, $,% usw.), IP-Adressen, Portnummern oder "http: // oder https: //" in Ihrem allgemeinen Namen verwenden. | wx.local |
| Email Adress | Geben sie hier die E-Mail-Adresse des Verantwortlichen an. | [wx@example.com](mailto:wx@example.com) |
| A challenge password | Optional |  |
| An optional company name | Optional |  |

#### Zertifikat ausstellen

Nachdem die Zertifikatsanfrage erstellt wurde, kann diese durch die CA verarbeitet werden. Das Zertifikat erhällt den Namen wx.pem.

pi@wx:~ $ openssl x509 -req -in wx.csr -CA ca.pem -CAkey caKey.pem -CAcreateserial -out wx.pem -days 36500 -sha512

#### PFX-Datei erzeugen

Für WebExpress wird eine PFX-Datei wx.pfx benötigt, welche die Zertifikatskette beinhaltet.

pi@wx:~ $ openssl pkcs12 -export -out wx.pfx -inkey wxKey.pem -in wx.pem -certfile ca.pem

Bei Ausführung des Befehles ist ein Passwort zu vergeben. Überprüfen Sie abschließend, ob die pfx-Datei korrekt ist.

pi@wx:~ $ openssl pkcs12 -info -in wx.pfx

### Zertifikate in WebExpress installieren

Für eine Https-Verbindung wird die pfx-Datei (wx.pfx) benötigt. Diese dient als Zertifikatsspeicher, indem alle relevanten Zertifikate enthalten sind. Diese muss auf dem Webserver übertragen und in der Webserverkonfiguration in das Verzeichnis /opt/wx/ssl hinterlegt werden. Hierfür muss zunächst jedoch das Verzeichnis erstellt werden.

pi@wx:~ $ sudo mkdir /opt/wx/ssl

Anschließend ist die pfx-Datei nach /opt/wx/ssl zu kopieren.

pi@wx:~ $ sudo cp wx.pfx /opt/wx/ssl

## WebExpress-Anwendungen installieren

WebExpress verfügt über ein leistungsstarkes Plugin-System. Die zu installierenden Plugins und gegebenenfalls Abhängigkeiten werden in das Verzeichnis /opt/wx kopiert. Eventuell muss das Plugin konfiguriert werden. Für die Installation und Einrichtung der Plugins sind die Anleitungen der Plugins heranzuziehen. Die Plugins werden erst aktiv, wenn WebExpress neu gestartet wurde.

pi@wx:~ $ sudo systemctl restart webexpress

# WebExpress starten

Für die erste Inbetriebnahme oder nach einer Änderung der Konfiguration ist WebExpress neu zu starten.

pi@wx:~ $ sudo systemctl restart webexpress

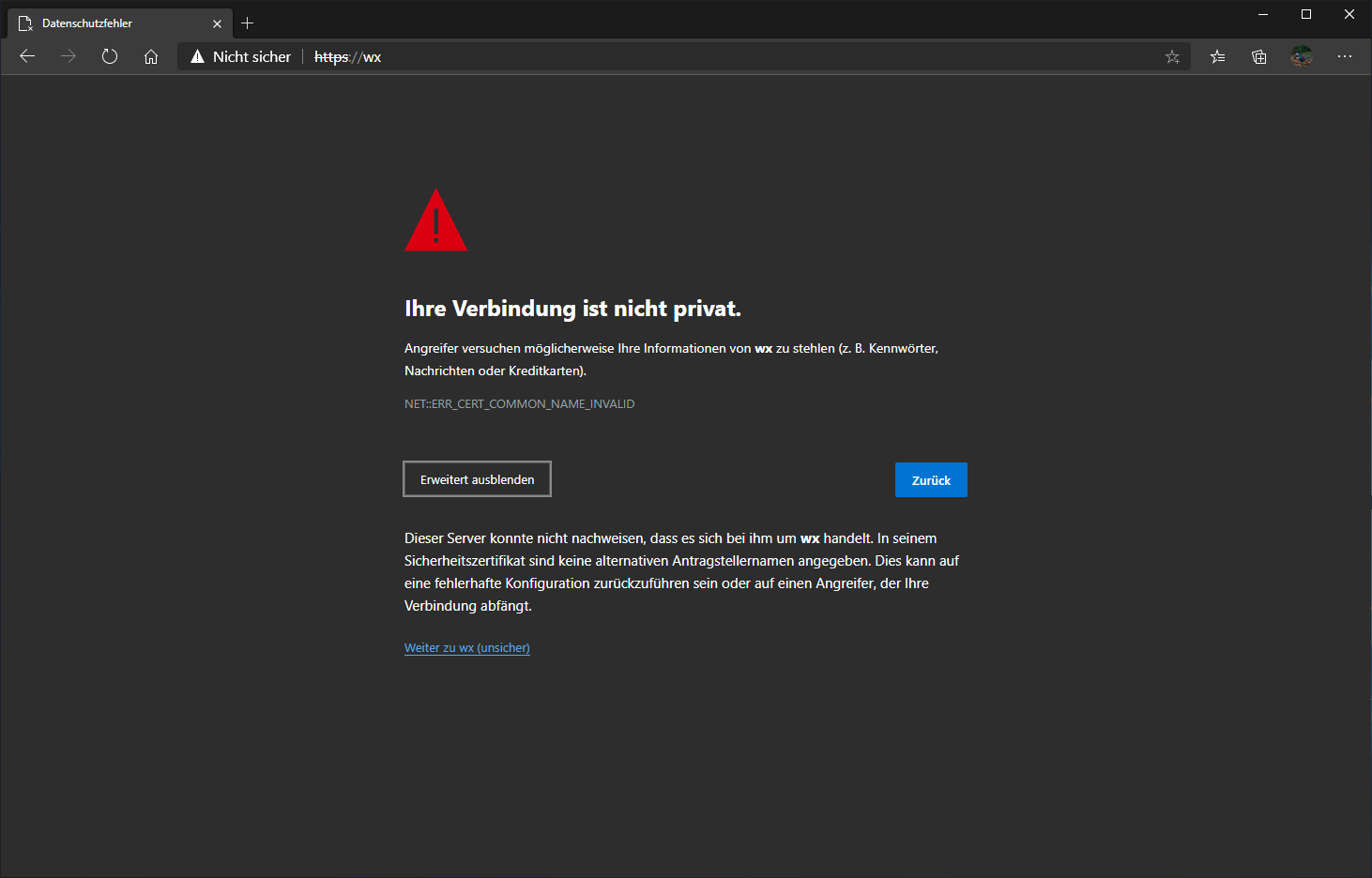
WebExpress wird nach jedem Neustarten der Rasperry Pis automatisch gestartet.

# Zertifikate in Windows installieren

Die pfx-Datei ist im Zertifikatsspeicher unter vertrauenswürdige Stammzertifizierungsstellen abzulegen.

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |

Im Browser ist das WebExpress-Zertifikat zu vertrauen.



# Update

Um die Sicherheit zu gewährleisten ist der Raspberry Pi, deren Anwendungen und WebExpress regelmäßig zu aktualisieren.

pi@wx:~ $ sudo raspi-config

pi@wx:~ $ sudo apt-get update

pi@wx:~ $ sudo apt-get upgrade

Die WebExpress-Binaries sind ebenfalls zu aktualisieren. Hierzu sind die aktuellen Binaries aus https://github.com/ReneSchwarzer/WebExpress/releases zu verwenden (siehe Abschnitt 3.5).

# Einkaufsliste

Als Hardware werden benötigt:

* Ein Raspberry Pi 4 Modell B mit 8GB
* Ein Steckernetzteil 5V/3A USB Typ-C
* Eine 16GB oder 32GB MicroSD Karte
* Optional ein Gehäuse

# Quellen

|  |  |
| --- | --- |
| [1] | <https://dotnet.microsoft.com/download/linux-package-manager/debian10/runtime-current> |
| [2] | <https://www.ionos.de/digitalguide/server/konfiguration/raspberry-pi-mit-fester-ip-adresse-versehen/#:~:text=Den%20Raspberry%20Pi%20mit%20einer%20festen%20IP-Adresse%20ausstatten.,Zeitraum%20mit%20anderen%20Ger%C3%A4ten%20auf%20ihn%20zugreifen%20will> |
| [3] | <https://legacy.thomas-leister.de/eine-eigene-openssl-ca-erstellen-und-zertifikate-ausstellen/> |