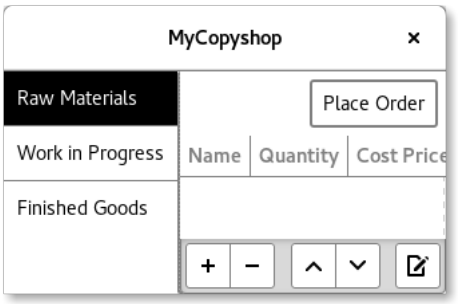

Wir wollen GNU Autotools verwenden, um auch größere Projekte leicht bauen, testen, verbreiten, etc. zu können.

Dazu sollte in einem neuen, leeren Projektordner ein Unterverzeichnis `src` oder `mycopyshop` oder Ähnliches mit z.B. einer noch leeren Datei namens `main.c` als Grundgerüst erzeugt werden. In diesem Ordner werden später alle Quelldateien des Programms stehen.

Nun wird im ursprünglichen Projektordner ein Terminal geöffnet und dort folgende Befehle ausgeführt:

```
$ git init
$ git add src/main.c
$ autoscan
```

Mit `git` kann leicht verwaltet werden, welche Dateien zum Projekt dazugehören. Alle fremden Dateien können mit `git clean -df` gelöscht werden. Mit `git status` wird der Zustand der Dateien im Projektordner angezeigt.

`autoscan` erzeugt eine GNU-Autoconf-Datei `configure.scan`, in welcher Projektname etc. einzutragen sind. Anschließend muss die fertige Datei in `configure.ac` umbenannt werden. Auch später kann `autoscan` noch ausgeführt werden, um Verbesserungsvorschläge für den Inhalt von `configure.ac` auszugeben.

Um damit nun ein Projekt zu konfigurieren wird üblicherweise eine Datei `autogen.sh` bereitgestellt, z.B. mit folgendem Inhalt:

```
#!/bin/sh
cd $(dirname $0)
autoreconf --force --install --verbose -Wall
```

Folgende Befehle machen die Datei ausführbar und führen sie danach aus:

```
$ chmod +x autogen.sh
$ ./autogen.sh
```

Nun kann das Projekt zwar automatisch konfiguriert werden, d.h. leicht an das Betriebssystem und vom Nutzer gewünschte Konfiguration z.B. des C-Compilers angepasst werden, aber noch nicht leicht automatisch gebaut werden. Dabei hilft GNU Automake:

```
$ automake --add-missing
```

Zudem sollte folgendes in `configure.ac` eingefügt werden, um Automake zu nutzen und darin Unterverzeichnisse benutzen zu können:

```
AC_CONFIG_FILES([Makefile])
AM_INIT_AUTOMAKE([-Wall -Werror -Wno-portability
                  subdir-objects])
```

Dies liefert zusammen mit `autogen.sh` die weiteren Instruktionen, um das nötige `Makefile.am` lauffähig zu machen.

Um darin anzugeben, dass ein Programm namens `copyshop` aus `src/main.c` und `src/cshopapp.c` erzeugt werden soll, sollte folgender Code ins `Makefile.am` eingefügt werden. Achtung, das Einrücken muss mit Tabulatoren geschehen.

```
AM_CPPFLAGS = -DDATADIR=\"$(datadir)\"
bin_PROGRAMS␣=␣src/copyshop
src_copyshop_SOURCES␣=␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣\
␣␣␣␣␣␣␣␣src/main.c␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣\
␣␣␣␣␣␣␣␣src/cshopapplication.c␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣\
␣␣␣␣␣␣␣␣src/cshopapplication.h␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣\
␣␣␣␣␣␣␣␣src/cshopwindow.c␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣\
␣␣␣␣␣␣␣␣src/cshopwindow.h
src_copyshop_LDADD␣=␣$(GTK_LIBS)
src_copyshop_CFLAGS␣=␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣\
␣␣␣␣␣␣␣␣-I$(top_srcdir)/␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣\
␣␣␣␣␣␣␣␣$(GTK_CFLAGS)
dist_data_DATA␣=␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣\
␣␣␣␣␣␣␣␣src/ui/menu.ui
```

Folgender Code in `configure.ac` definiert die nötigen Variablen `GTK_LIBS` und `GTK_CFLAGS`:

```
# Checks for libraries.
PKG_CHECK_MODULES([GTK],[gtk+-3.0 >= 3.16])
```

Auf die Konfiguration kann zugegriffen werden mit
`#include <config.h>`.

Nun kann das Programm wie folgt mit den Standardoptionen konfiguriert und kompiliert, installiert und wieder deinstalliert werden:

```
$ ./autogen.sh
$ CFLAGS="-Wall" CPPFLAGS="-Wall" ./configure
$ make
# make install
# make uninstall
```

Folgender Befehl erzeugt eine Datei mit allem Quellcode („Tarball“) zur Distribution:

```
$ make dist
```

Bei ernsthaften Projekten bietet es sich an, die Handbücher der GNU Autotools nach weiteren Empfehlungen zu durchsuchen:

- `https://www.gnu.org/software/autoconf/manual/autoconf.html`
- `https://www.gnu.org/software/automake/manual/automake.html`