

# Zarządzanie projektami C/C++

- Problemy do omówienia:
  - jak utworzyć projekt dla kodu w C/C++
  - podstawowe mechanizmy zarządzania projektami (importowanie)

- Projekt C/C++:
  - pojemnik na kod źródłowy C/C++
    - różne typy projektów
  - zanim rozpoczniemy pisanie kodu C/C++ w QNX<sup>®</sup> IDE
    - musimy utworzyć projekt który będzie zawierał kod źródłowy

- Budowanie z użyciem “make” i “Makefiles”:
  - narzędzie linii poleceń **make** jest wykorzystywane przez IDE do kontroli procesu budowania plików wykonawczych i bibliotek
  - **make** wykorzystuje Makefile:
    - może zawierać komendy sterujące procesem budowania (np. wywołania kompilatora i linkera)
    - opisuje zależności w drzewie źródeł
      - np. xxx.o potrzebuje być przebudowany jeżeli xxx.c i xxx.h zmienią się change
      - wykorzystanie daty i czasu modyfikacji podczas budowania
      - np. jeśli data i czas utworzenia xxx.c lub xxx.h jest nowszy niż xxx.o wtedy buduj nowy xxx.o
  - w celu uzyskania informacji o Makefiles patrz **make** w pomocy ‘Utilities Reference’

- Projekty dla C lub C++:
  - Standard Make C
  - Standard Make C++
  - QNX C Application
  - QNX C++ Application
  - QNX C Library
  - QNX C++ Library

Który z nich użyć? ...

- Projekty ‘Standard Make’ oraz ‘QNX projects’:
  - Projekty Standard Make C/C++:
    - bardziej elastyczne sterowanie procesem budowania
    - łatwiejsze importowanie istniejących, niebanalnych, drzew źródeł
    - możliwość wykorzystywania innych niż **make** narzędzi budowania
  - Projekty QNX C/C++ :
    - łatwość budowania na wiele platform procesorowych, np. rozwój oprogramowania dla PPC i x86
    - nie musimy pisać własnego skryptu Makefile

- Tematy:

- Przegląd**

- Projekty 'Standard Make C/C++'**

- Projekty 'QNX C/C++ Application'**

- Importowanie kodu źródłowego**

- Ćwiczenie**

- Podsumowanie**

- Projekt 'Standard Make C/C++':
  - pusty projekt do którego możemy dodać cokolwiek chcemy
  - jest ogólnym projektem, nie jest specyficzny dla QNX
  - domyślnie, kiedy chcemy zbudować projekt wywoływana jest komenda:  
`make -k all`  
jak zobaczymy, może być zmieniona na dowolną komendę budowania
  - ten typ projektu jest użyteczne w sytuacji:
    - jeżeli posiadamy istniejącą strukturę budowania i/lub narzędzia i chcemy to zachować
    - chcemy budować projekt nie korzystając ze środowiska IDE



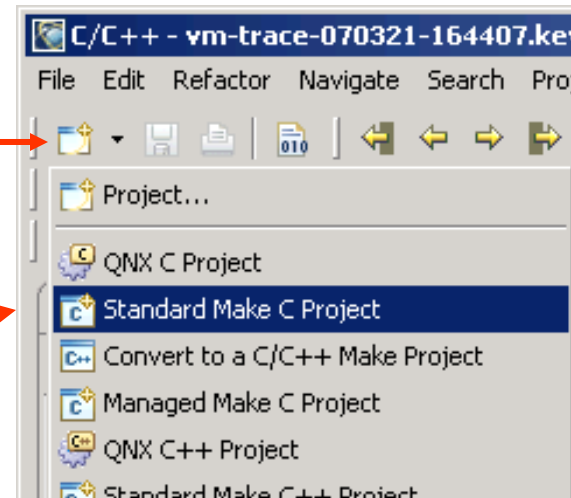
- Tworzenie projektu:
  - otwieramy perspektywę ‘C/C++ Development’

1

idziemy do menu ‘New’

2

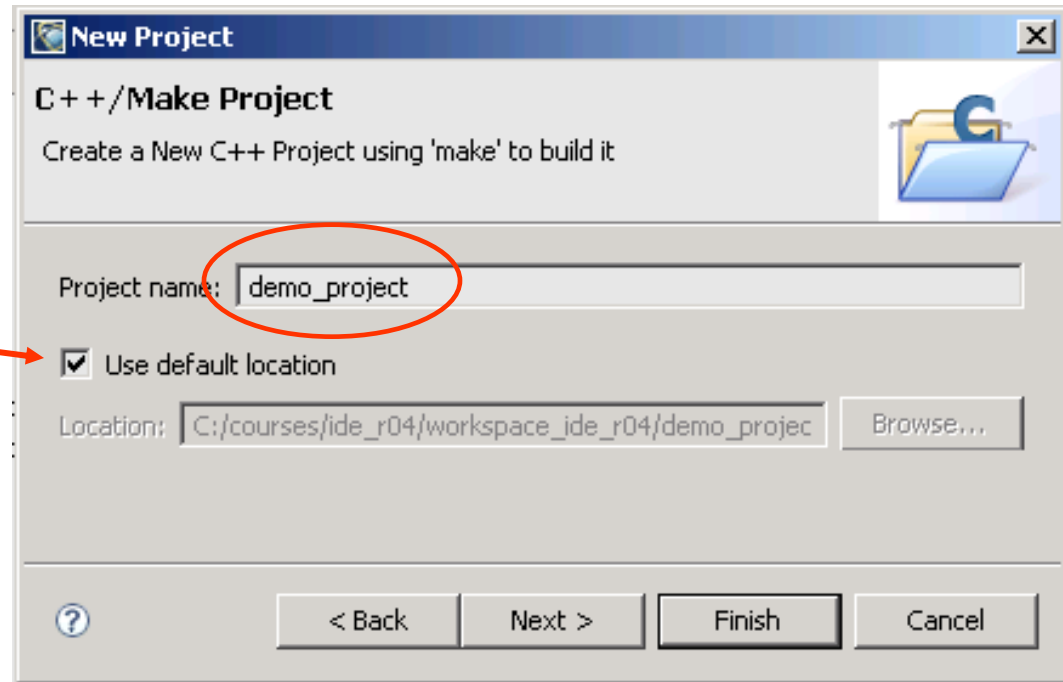
wybieramy właściwy typ projektu



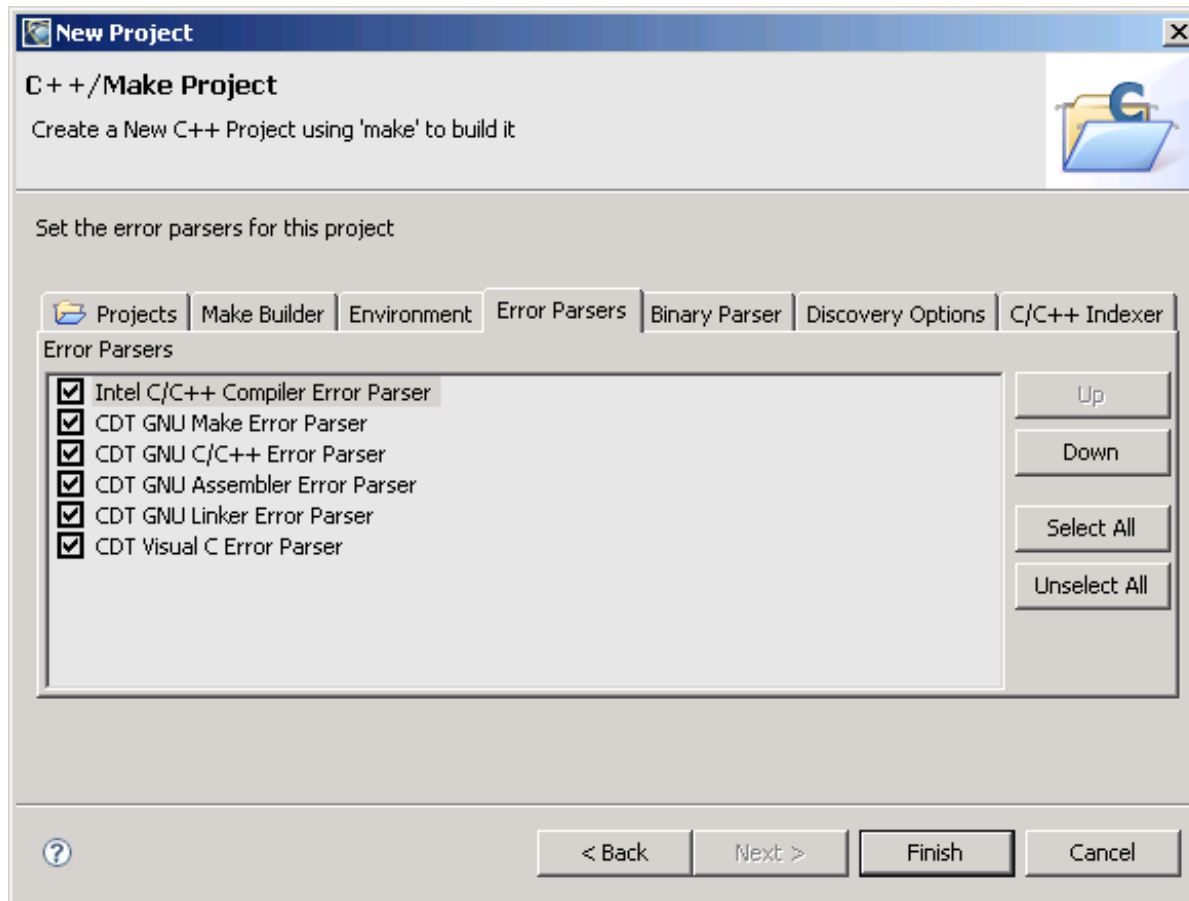
- powinien pojawić się kreator tworzenia projektu

- Wpisujemy odpowiednią nazwę:

domyślna lokalizacja  
jest wewnątrz katalogu  
roboczego, odznaczenie  
tego wyboru pozwala  
umieścić projekt w  
dowolnym miejscu systemu  
plików



- Mamy do dyspozycji wiele ustawień:



– ale zwykle pozostawiamy wartości domyślne

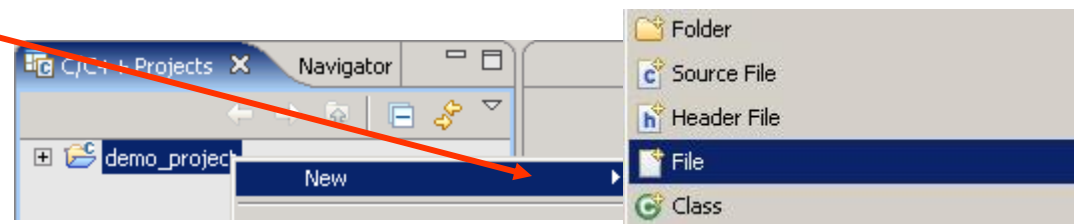
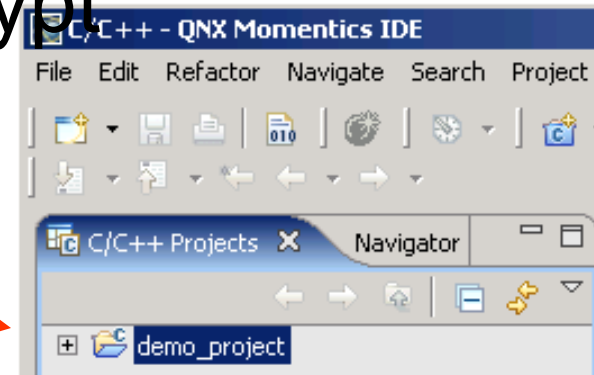


- Następnie tworzymy skrypt

- Makefile:

- niedawno utworzony projekt
- jeżeli to rozwiniemy, zobaczymy że jest pusty
- teraz możemy go wypełnić
  - jeżeli posiadamy istniejącą strukturę źródeł, możemy ją skopiować do wewnątrz projektu,
  - w przeciwnym razie, musimy utworzyć pliki, zaczynając od Makefile...

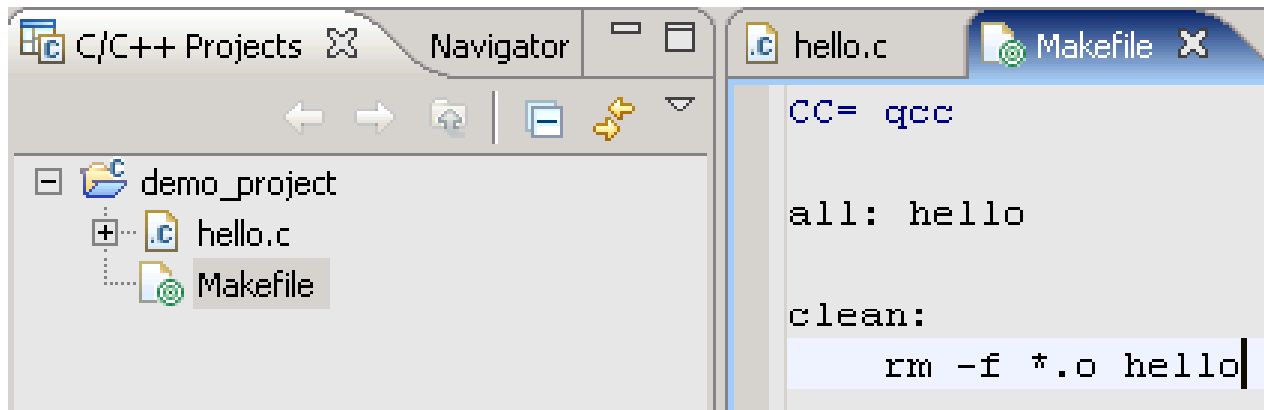
wybieramy projekt,  
prawy-klik,  
wybieramy New->File



- wykonujemy wszystkie kroki w kreatorze celem utworzenia pliki
- minimalna zawartość skryptu Makefile to...



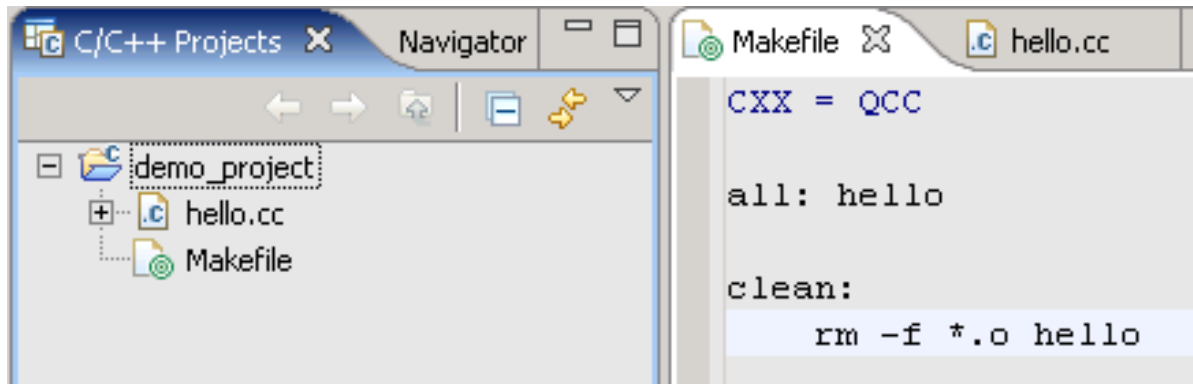
- Minimalna zawartość Makefile projektu C:
  - minimalnie Makefile może zawierać:



zapis ten zakłada, że będziemy edytować plik źródłowy **hello.c**

# Projekty 'Standard Make C/C++' – Wypełnianie projektu

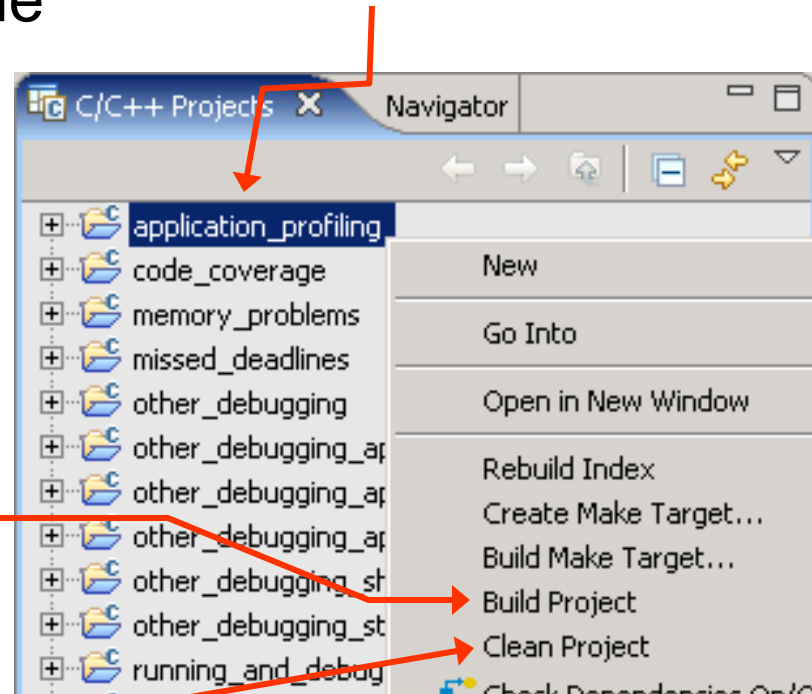
- Minimalna zawartość Makefile projektu C++:
  - minimalnie Makefile może zawierać :



zapis ten zakłada, że będziemy edytować plik źródłowy **hello.cc**

- więcej informacji o skryptach Makefiles w **make** w Utilities Reference

- Budowanie pliku wykonawczego:
  - prawy-klik na projekcie



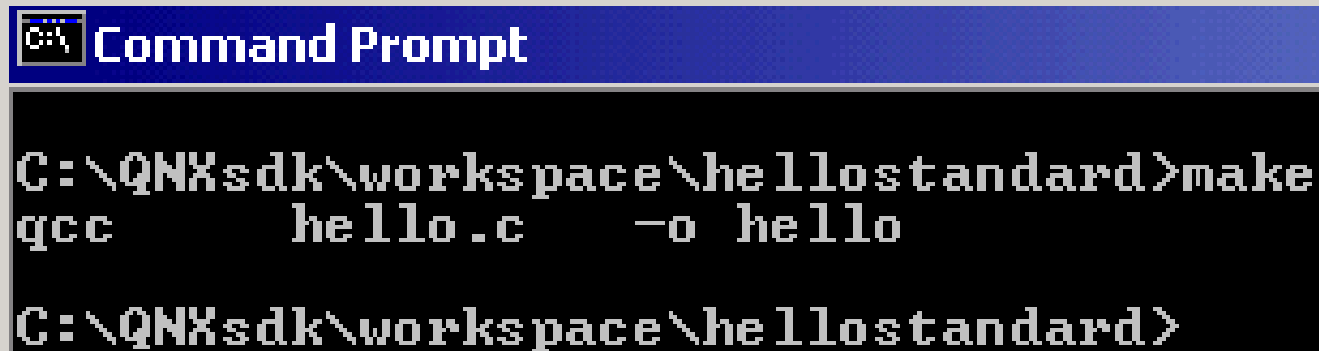
budowanie tylko tych elementów,  
których kod uległ zmianie  
(np. `make -k all`)

usunięcie: wykonawczych, obiektów, dzienników błędów, itp  
(np. `make -k`)





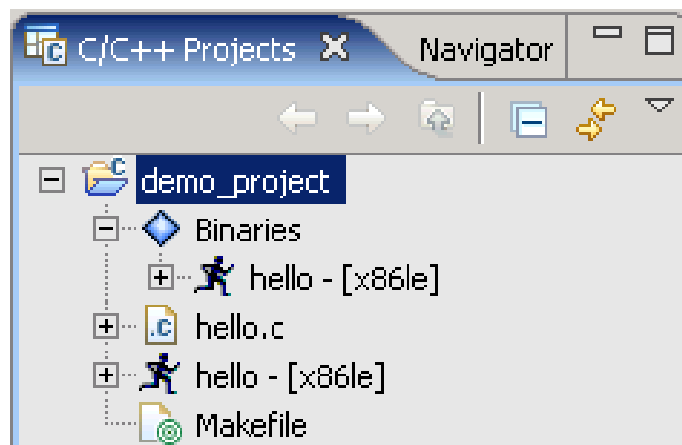
- Budowanie pliku wykonawczego – poza środowiskiem IDE:
  - projekt jest po prostu katalogiem, zawierającym pliki
  - możemy zbudować projekt z linii poleceń:



The screenshot shows a Windows Command Prompt window with a blue title bar that reads "Command Prompt". The window has a black background with white text. The command prompt shows the current directory as `C:\QNXsdk\workspace\hellostandard`. The user enters the command `make`, which outputs `gcc hello.c -o hello`. The prompt then returns to `C:\QNXsdk\workspace\hellostandard>`.

```
C:\QNXsdk\workspace\hellostandard>make
gcc      hello.c      -o hello
C:\QNXsdk\workspace\hellostandard>
```

- Nasz zbudowany projekt:



- katalog '**Binaries**' faktycznie nie istnieje
  - Jest tutaj po to, żeby pokazać wszystkie pliki wykonawcze z tego projektu w jednym miejscu.
  - Jeżeli mamy wiele plików wykonawczych lub skomplikowaną strukturę katalogów, może to być bardzo pomocne

- Tematy:

**Przegląd**

**Projekty 'Standard Make C/C++'**

**→ Projekty 'QNX C/C++ Application'**

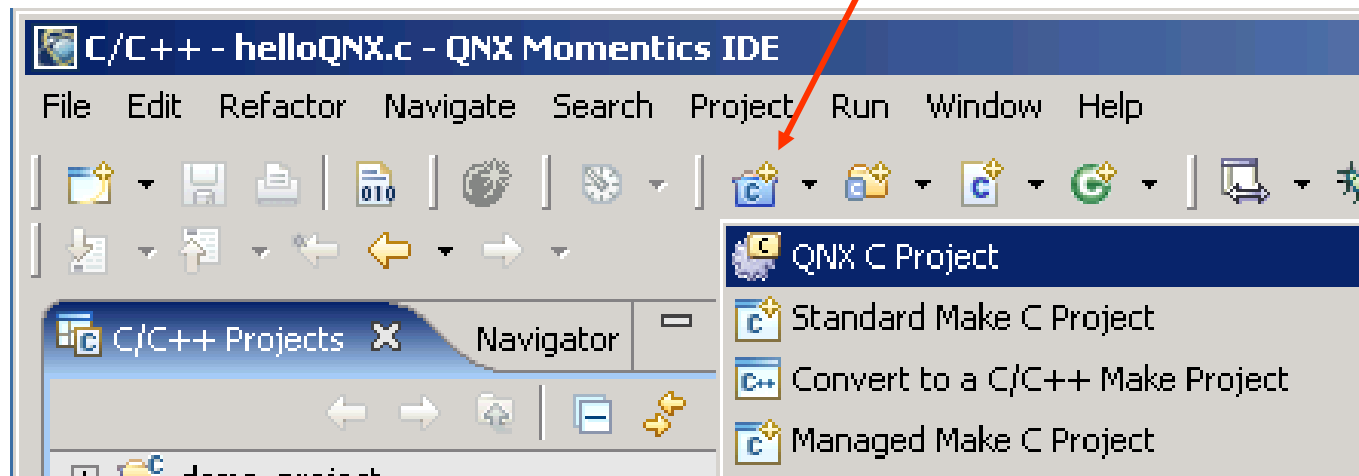
**Importowanie kodu źródłowego**

**Ćwiczenie**

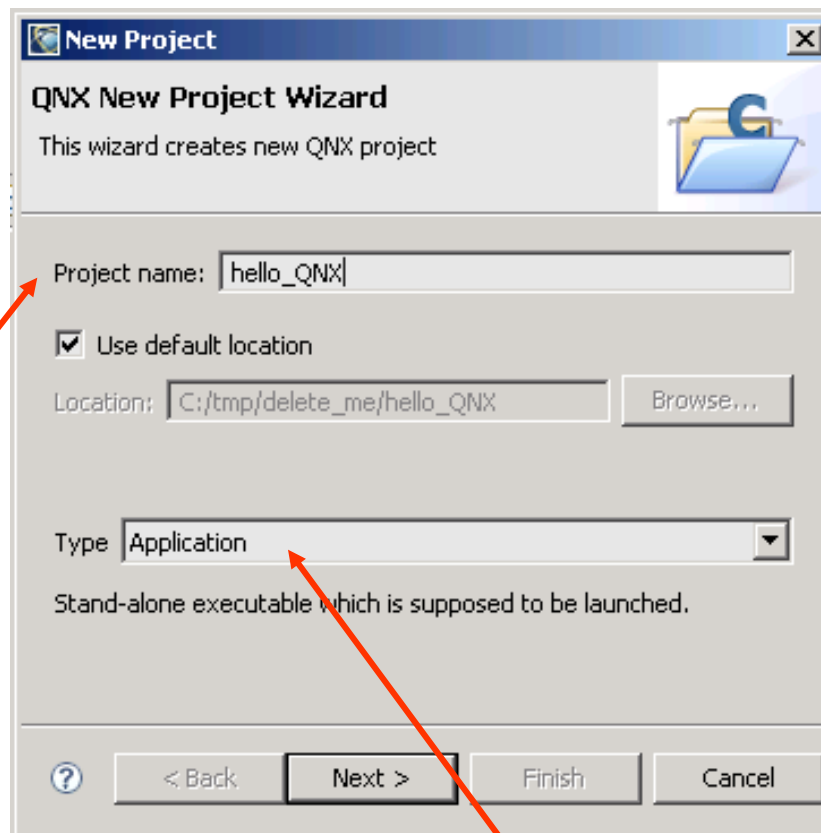
**Podsumowanie**

- Projekt 'QNX C/C++ Application':
  - jest to projekt:
    - ze strukturą Makefile przeznaczoną dla QNX
    - wspiera wiele wariantów/CPU/platform

- W celu utworzenia projektu:
  - rozpoczynamy tak jak powiedzieliśmy dla projektu 'standard Make'
  - lub z menu 'New C/C++ Project':

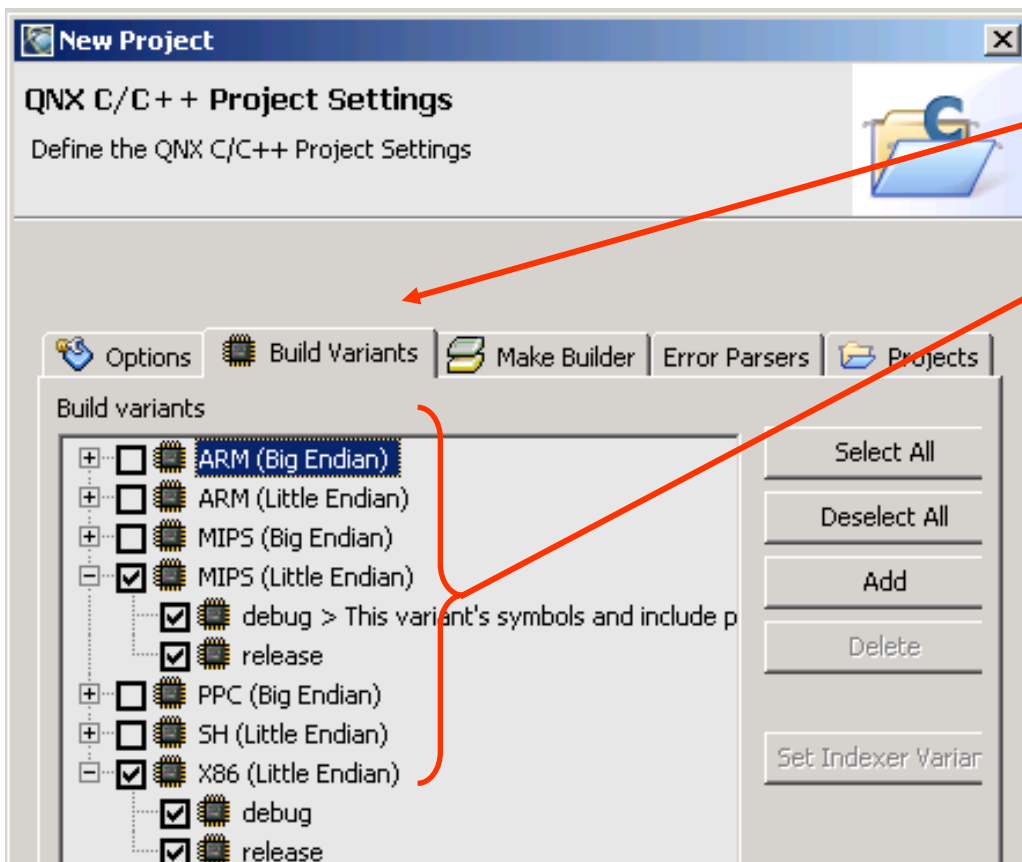


- W celu utworzenia projektu:



- 1 wprowadzamy nazwę
- 2 wybieramy 'Application', wśród innych możliwości, jak np. biblioteki
- 3 wybieramy 'Next'

- W celu utworzenia projektu:

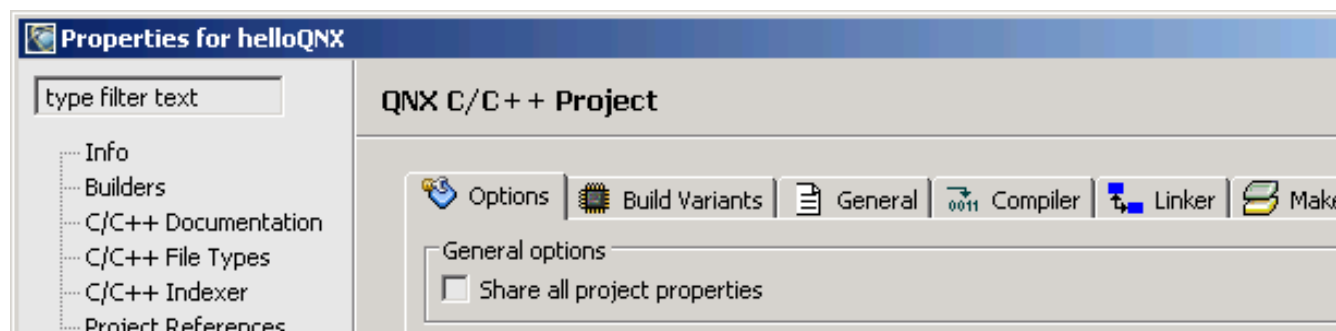
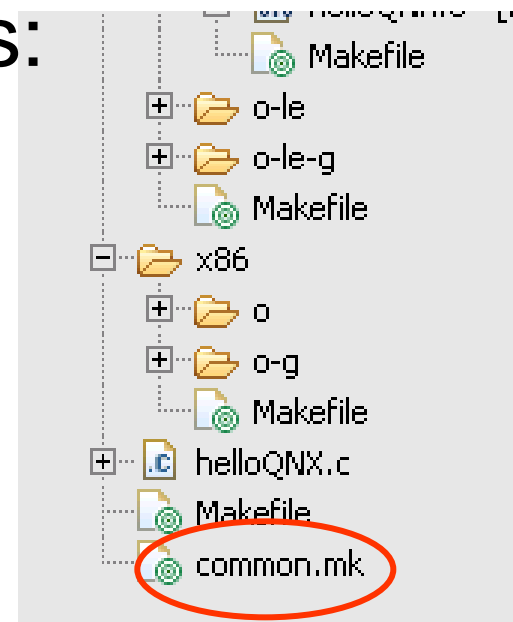


1 wybieramy odpowiednie warianty budowania włączając wybór wersji debug i/lub końcowej

2 wybieramy 'Finish' i nasz projekt jest utworzony

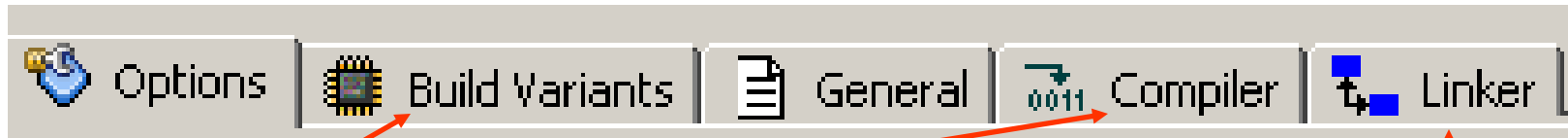
- Praca ze skryptami Makefiles:

- nie modyfikujemy pliku Makefiles
- Makefiles dołącza `common.mk`
- możemy modyfikować `common.mk`, upewniwszy się wcześniej czy jest taka opcja we właściwościach projektu





- Opcje budowania:

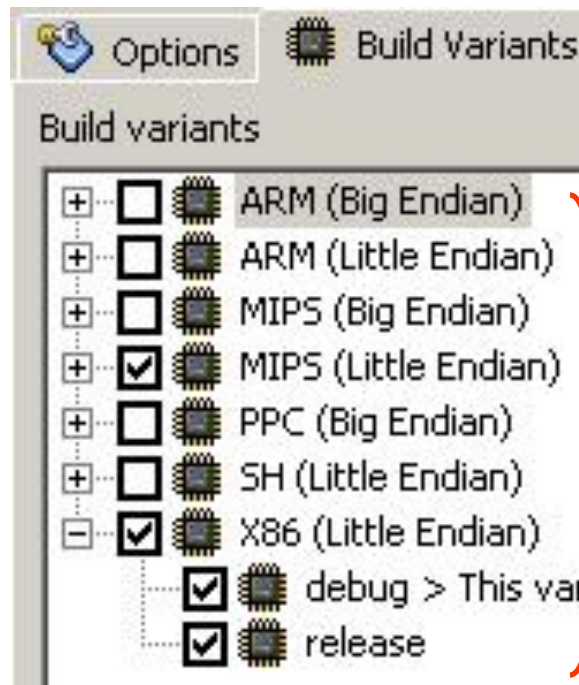


Zmiana architektury CPU i wersji  
release i/lub debug

Opcje kompilatora, np.

- lokalizacja źródeł, położonych na zewnątrz projektu
- lokalizacja plików nagłówkowych, położonych w innych niż domyślne lokalizacjach
- poziom optymalizacji, poziom ostrzeżeń

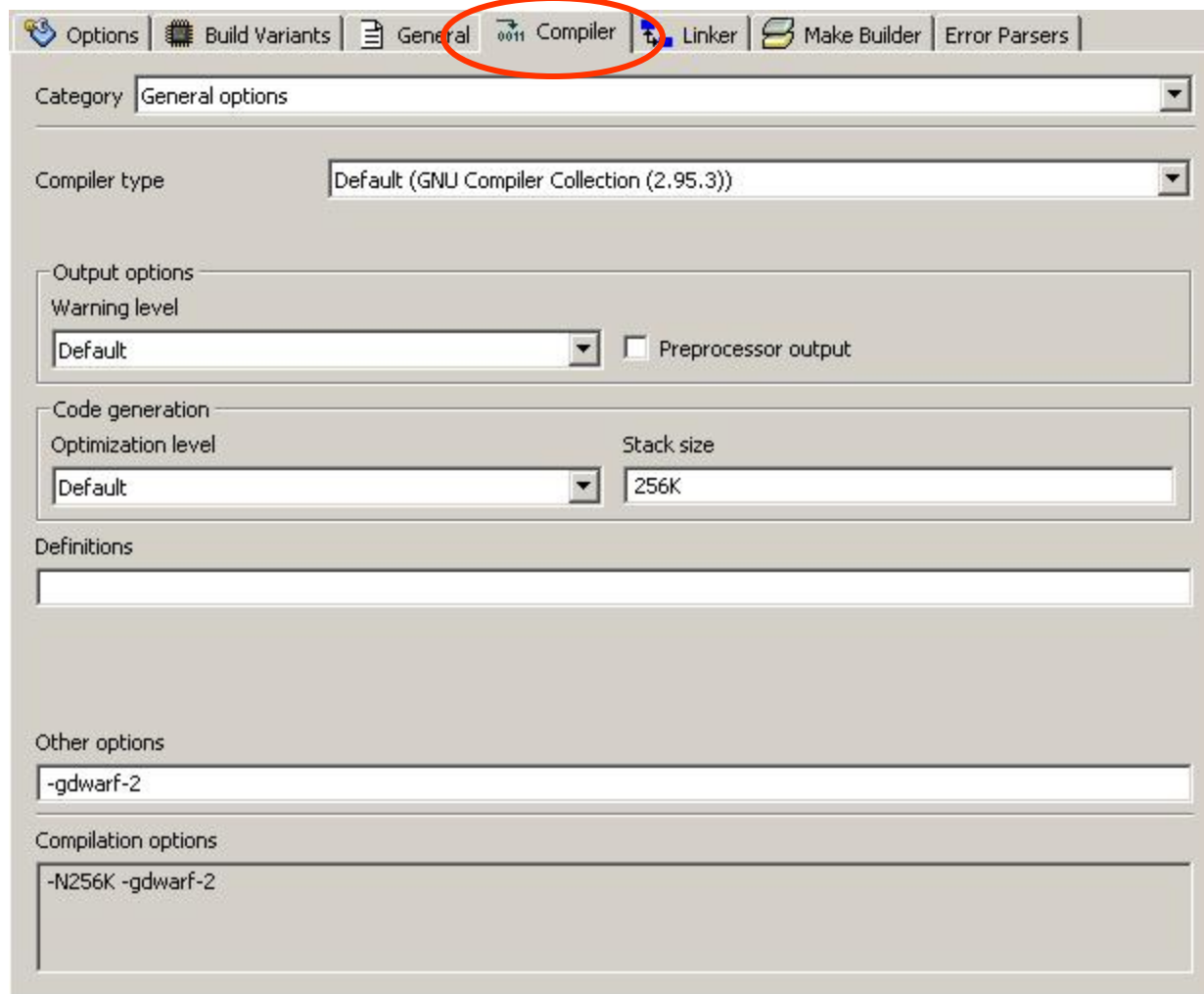
Opcja linkera,  
np. dodatkowe biblioteki



wybór skutkuje w  
wyniku końcowym  
budowania

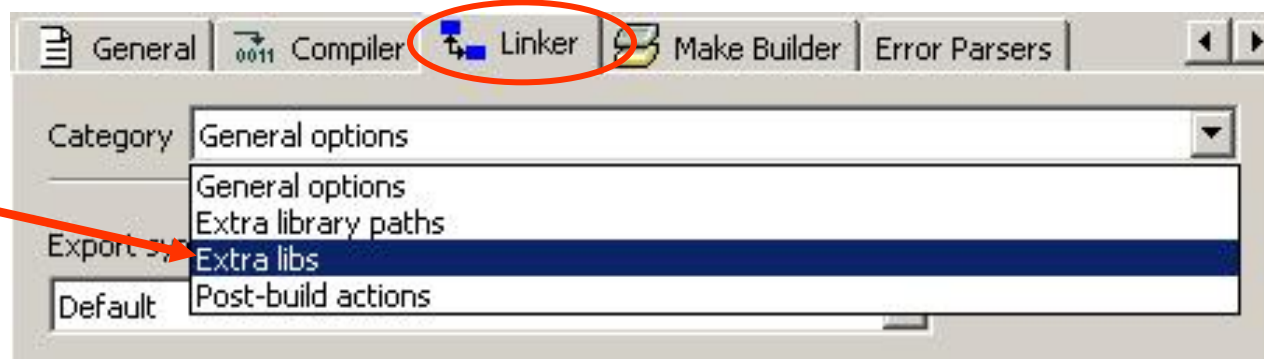


- Ustawianie opcji kompilacji:



- Dodanie biblioteki matematycznej:
  - w panelu linkera

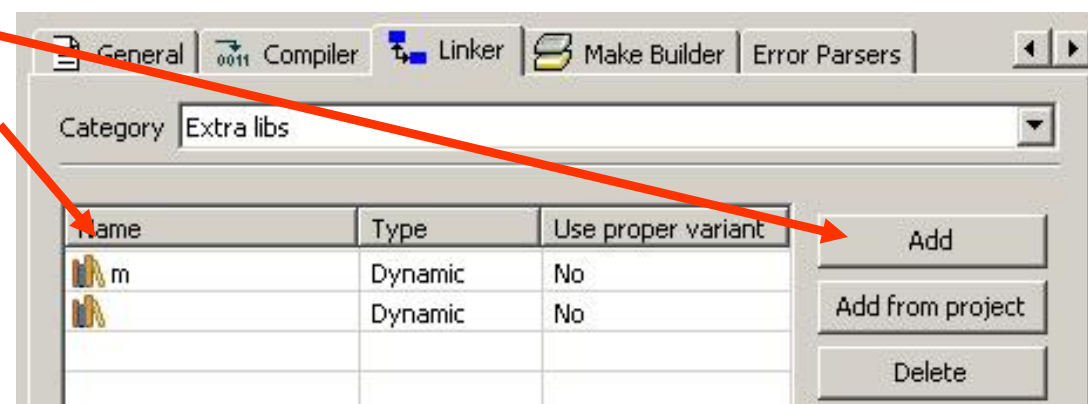
wybór kategorii



klikamy 'Add'

Wprowadzamy nazwę biblioteki:

wprowadzamy tylko część (np. dla **libm.a**, wprowadzamy **m**)



- Budowanie wykonawczego – z poziomu IDE:

- aby zbudować oprogramowanie w IDE, prawy klik na projekcie...

budowanie z wybranym profilem

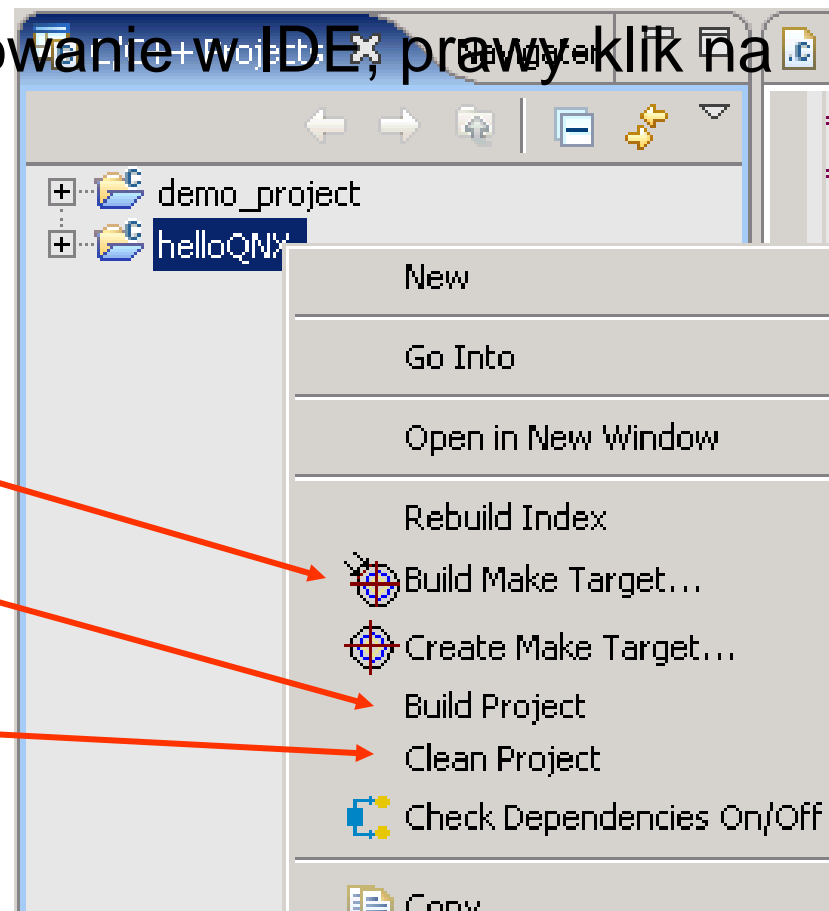
```
make -k your_target
```

budowanie tylko elementów  
wymagających tego

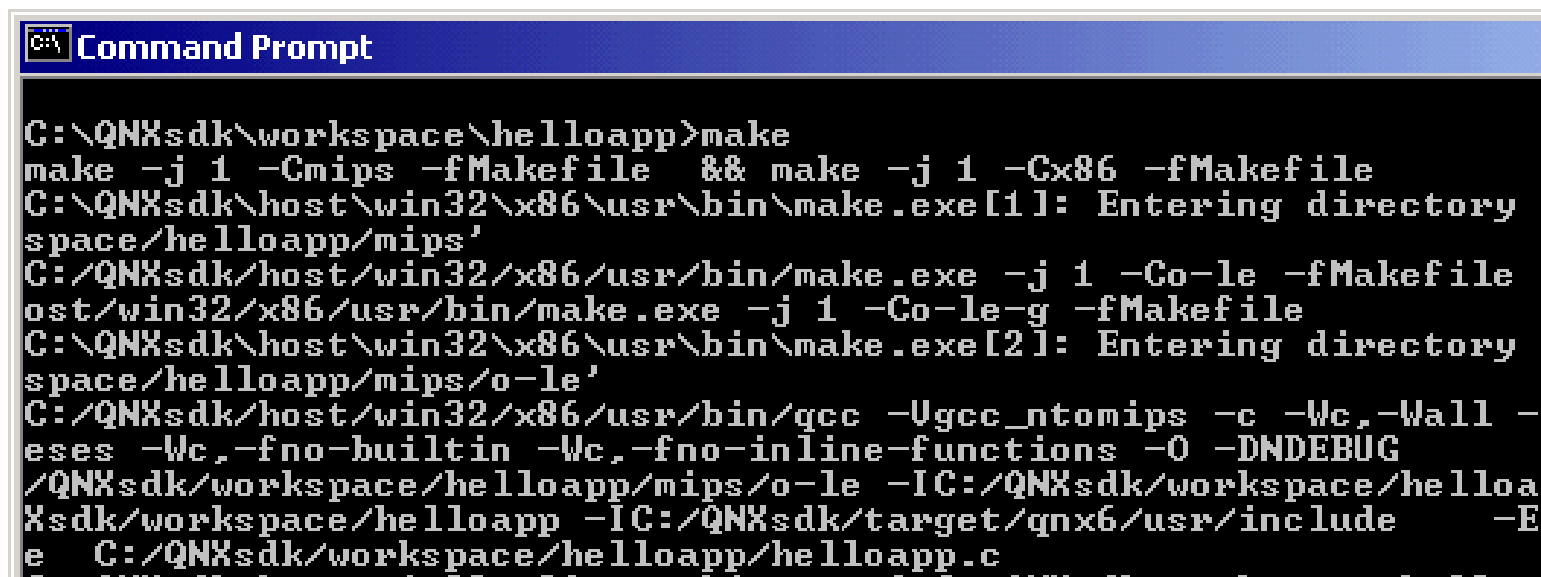
```
make -k all
```

usunięcie wszystkich plików nie będących  
źródłami (np. wykonawcze, obiekty, pliki  
błędów)

```
make -k clean
```



- Budowanie wykonawczego - poza IDE:
  - projekt jest po prostu katalogiem, zawierającym pliki
  - możemy zbudować projekt z poziomu linii poleceń



```
Command Prompt

C:\QNXsdk\workspace\helloapp>make
make -j 1 -Cmips -fMakefile  && make -j 1 -Cx86 -fMakefile
C:\QNXsdk\host\win32\x86\usr\bin\make.exe[1]: Entering directory
space/helloapp/mips'
C:/QNXsdk/host/win32/x86/usr/bin/make.exe -j 1 -Co-le -fMakefile
ost/win32/x86/usr/bin/make.exe -j 1 -Co-le-g -fMakefile
C:\QNXsdk\host\win32\x86\usr\bin\make.exe[2]: Entering directory
space/helloapp/mips/o-le'
C:/QNXsdk/host/win32/x86/usr/bin/gcc -Ugcc_ntomips -c -Wc,-Wall -
eses -Wc,-fno-builtin -Wc,-fno-inline-functions -O -DNDEBUG
/QNXsdk/workspace/helloapp/mips/o-le -IC:/QNXsdk/workspace/helloa
Xsdk/workspace/helloapp -IC:/QNXsdk/target/qnx6/usr/include -E
e C:/QNXsdk/workspace/helloapp/helloapp.c
```

- Tematy:

- Przegląd**

- Projekty 'Standard Make C/C++'**

- Projekty 'QNX C/C++ Application'**

- Importowanie kodu źródłowego**

- Ćwiczenie**

- Podsumowanie**

- Istnieje kilka sposobów na umieszczenie kodu źródłowego w IDE:
  - system kontroli źródeł
  - kreator importu IDE
  - wykorzystanie kodu położonego poza środowiskiem roboczym
  - korzystanie z zewnętrznych narzędzi kopiowania

- Tematy:

- Przegląd**

- Projekty 'Standard Make C/C++'**

- Projekty 'QNX C/C++ Application'**

- Importowanie kodu źródłowego**

- Ćwiczenie**

- Podsumowanie**



- Tworzenie projektu typu 'Standard Make C/C++':
  - utworzyć 'Standard Make C Project' lub 'Standard Make C++ Project'. Nazwać go **standard**
  - dodać dwa nowe pliki:
    - plik źródłowy z kodem funkcji *main()*
    - Makefile – wcześniejsze slajdy
  - zbudować projekt aby upewnić się, że wszystko jest dobrze
- Tworzenie projektu typu 'QNX C/C++ Project'.

- Omawiane tematy:
  - tworzenie projektów C/C++
    - Standard Make C/C++ Projects
    - QNX C/C++ Application Projects
  - podstawy zarządzania projektami
  - umieszczanie kodu źródłowego wewnątrz IDE