CODE::BLOCKS & VALGRIND

OPRACOWAŁ

MICHAŁ BETHKE

O CZYM PREZENTACJA?

- ✓ **Code::Blocks**
  - ✓ Informacje wstępne
  - ✓ Skąd ściągnąć? Jak zainstalować? (wersja linuksowa)
  - ✓ Rzut okiem na panel główny
  - ✓ Tworzenie naszego pierwszego projektu
  - ✓ Debugger
- ✓ **Valgrind**
  - ✓ Instalacja/konfiguracja
  - ✓ GCC a Valgrind
  - ✓ Składnia
  - ✓ Funkcjonalności
    - ✓ Wykrywanie wycieków pamięci
    - ✓ Wykrywanie odniesień poza pamięć
    - ✓ Wykrywanie niezainicjalizowanych zmiennych
    - ✓ Inne

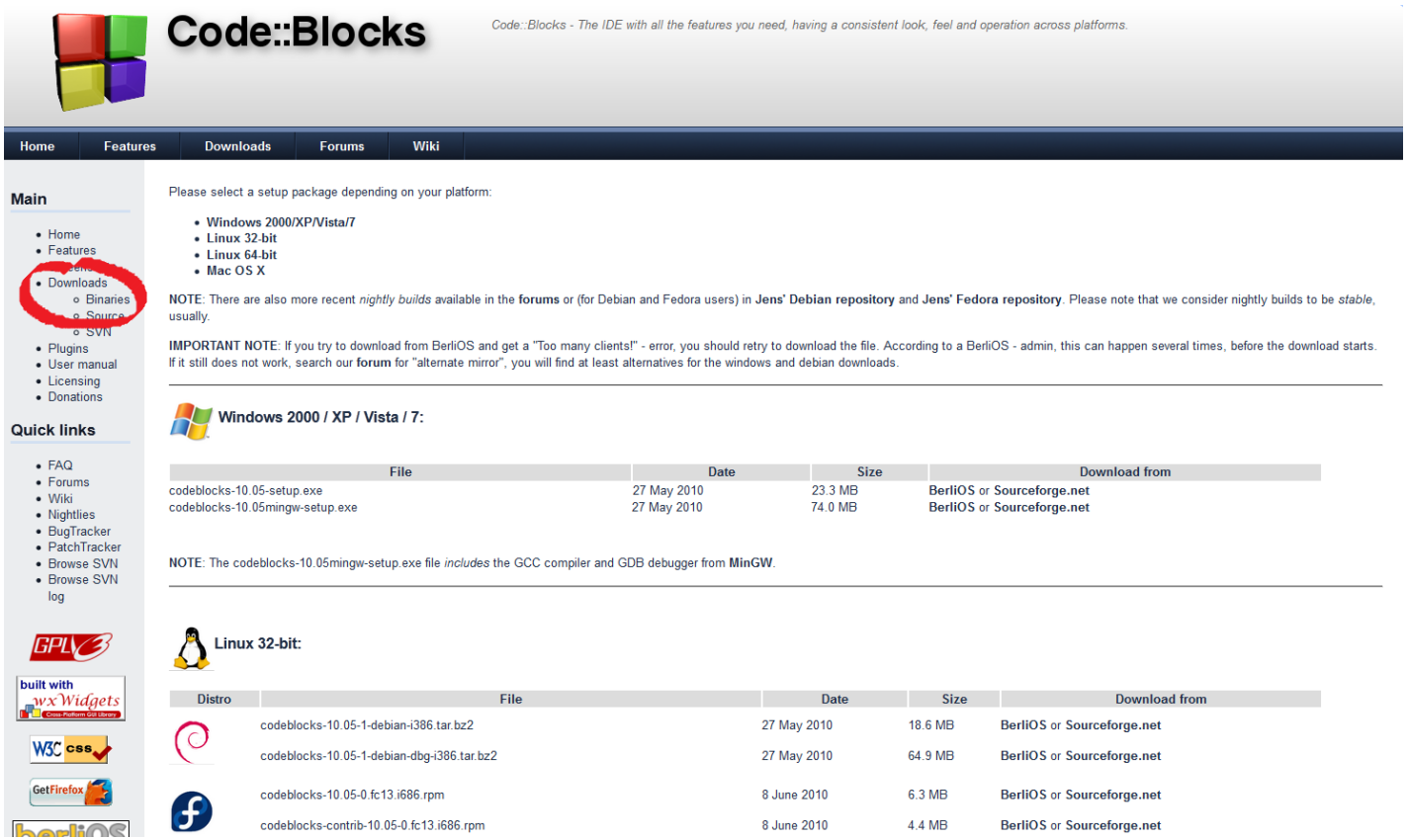
CODE::BLOCKS

INFORMACJE WSTĘPNE

Code::Blocks - zintegrowane środowisko programistyczne na licencji GNU (a więc całkowicie darmowe), dla programistów C/C++. Cechuje je niska pamięciożerność oraz prostota obsługi. Działa zarówno pod Windowsem, jak i Linuksem.

SKĄD ŚCIĄGNĄĆ? JAK ZAINSTALOWAĆ?

Wchodzimy na WWW.CODEBLOCKS.ORG – ściągamy!



Code::Blocks

Code::Blocks - The IDE with all the features you need, having a consistent look, feel and operation across platforms.

Home | Features | Downloads | Forums | Wiki

Main

- Home
- Features
- Downloads
  - Binaries
  - Source
  - SVN
- Plugins
- User manual
- Licensing
- Donations

Quick links

- FAQ
- Forums
- Wiki
- Nightlies
- BugTracker
- PatchTracker
- Browse SVN
- Browse SVN log

Please select a setup package depending on your platform:

- **Windows 2000/XP/Vista/7**
- **Linux 32-bit**
- **Linux 64-bit**
- **Mac OS X**

NOTE: There are also more recent *nightly builds* available in the **forums** or (for Debian and Fedora users) in **Jens' Debian repository** and **Jens' Fedora repository**. Please note that we consider nightly builds to be *stable*, usually.

IMPORTANT NOTE: If you try to download from BerliOS and get a "Too many clients!" - error, you should retry to download the file. According to a BerliOS - admin, this can happen several times, before the download starts. If it still does not work, search our **forum** for "alternate mirror", you will find at least alternatives for the windows and debian downloads.

Windows 2000 / XP / Vista / 7:

File	Date	Size	Download from
codeblocks-10.05-setup.exe	27 May 2010	23.3 MB	**BerliOS** or **Sourceforge.net**
codeblocks-10.05mingw-setup.exe	27 May 2010	74.0 MB	**BerliOS** or **Sourceforge.net**

NOTE: The codeblocks-10.05mingw-setup.exe file *includes* the GCC compiler and GDB debugger from **MinGW**.

Linux 32-bit:

Distro	File	Date	Size	Download from
	codeblocks-10.05-1-debian-i386.tar.bz2	27 May 2010	18.6 MB	**BerliOS** or **Sourceforge.net**
	codeblocks-10.05-1-debian-dbg-i386.tar.bz2	27 May 2010	64.9 MB	**BerliOS** or **Sourceforge.net**
	codeblocks-10.05-0.fc13.i686.rpm	8 June 2010	6.3 MB	**BerliOS** or **Sourceforge.net**
	codeblocks-contrib-10.05-0.fc13.i686.rpm	8 June 2010	4.4 MB	**BerliOS** or **Sourceforge.net**

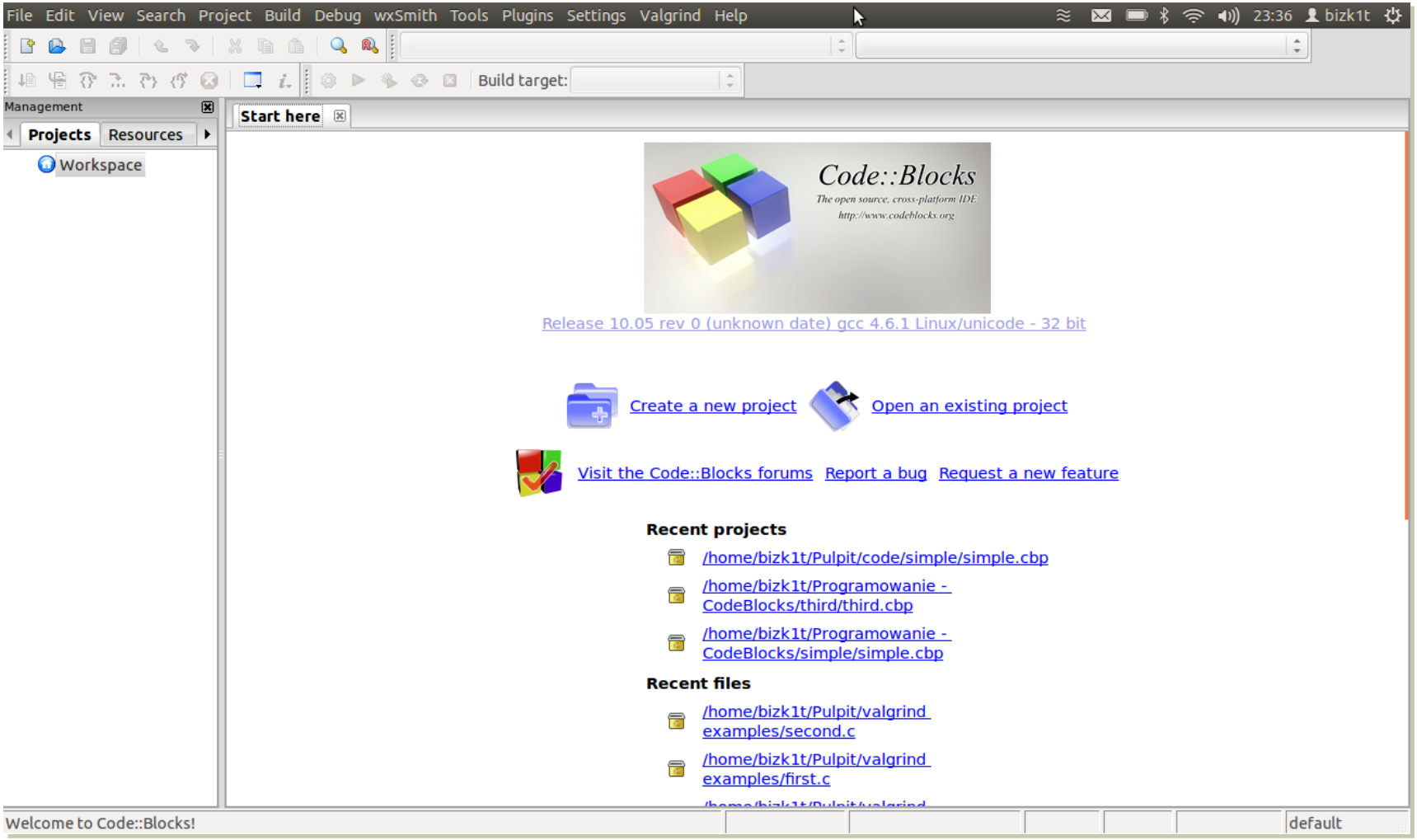
lub…

...w przypadku linuksów, w konsoli wpisujemy:

```
sudo apt-get/yum install codeblocks
```

```
bizk1t@bizk1t-laptop:~/Pulpit$ sudo apt-get install codeblocks
[sudo] password for bizk1t:
Czytanie list pakietów... Gotowe
Budowanie drzewa zależności
Odczyt informacji o stanie... Gotowe
Sugerowane pakiety:
  libwxgtk2.8-dev wx-common codeblocks-contrib
Zostaną zainstalowane następujące NOWE pakiety:
  codeblocks
0 aktualizowanych, 1 nowo instalowanych, 0 usuwanych i 1 nieaktualizowanych.
Konieczne pobranie 0 B/1642 kB archiwów.
Po tej operacji zostanie dodatkowo użyte 4686 kB miejsca na dysku.
Selecting previously unselected package codeblocks.
(Odczytywanie bazy danych ... 236057 files and directories currently installed.)
Rozpakowanie codeblocks (z .../codeblocks_10.05-2_i386.deb) ...
Przetwarzanie wyzwalaczy dla shared-mime-info...
Przetwarzanie wyzwalaczy dla desktop-file-utils...
Przetwarzanie wyzwalaczy dla bamfdaemon...
Rebuilding /usr/share/applications/bamf.index...
Przetwarzanie wyzwalaczy dla gnome-menus...
Przetwarzanie wyzwalaczy dla man-db...
Przetwarzanie wyzwalaczy dla menu...
Konfigurowanie codeblocks (10.05-2) ...
Przetwarzanie wyzwalaczy dla menu...
bizk1t@bizk1t-laptop:~/Pulpit$
```

RZUT OKIEM NA PANEL GŁÓWNY

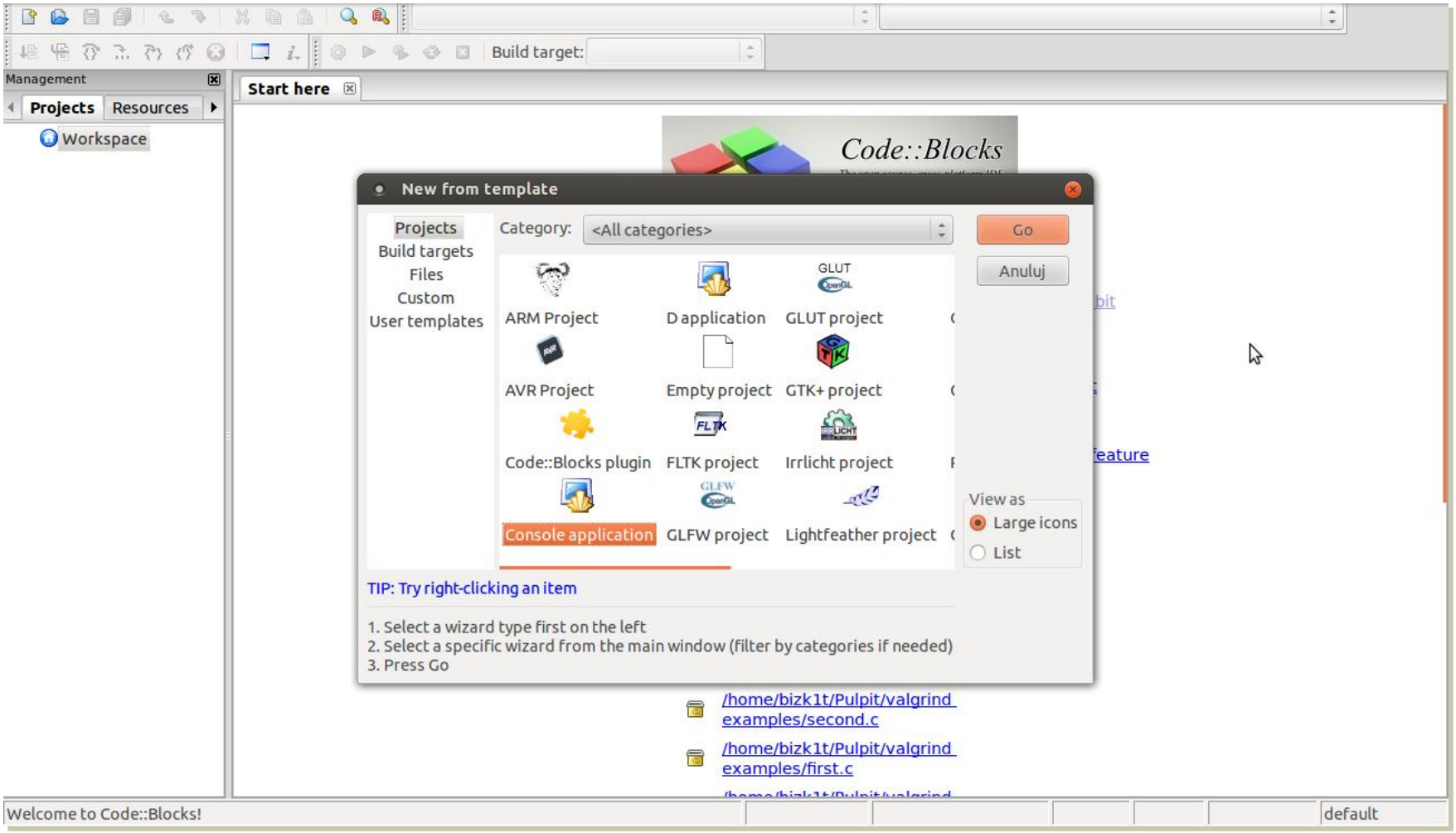


TWORZENIE NASZEGO PIERWSZEGO PROJEKTU

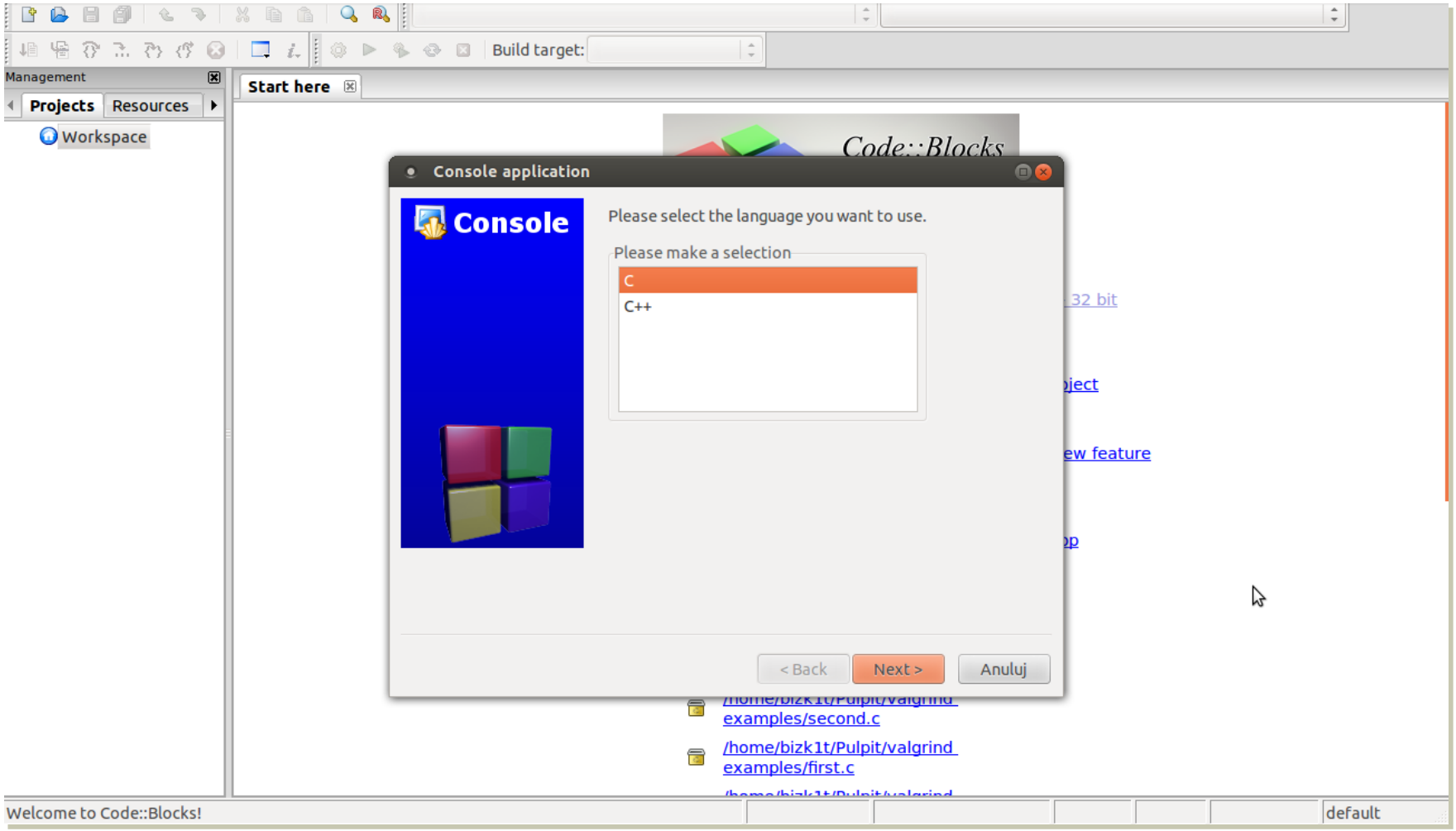
Standardowo jak większość IDE, Code::Blocks oferuje wygodne tworzenie większych projektów, składających się z wielu plików nagłówkowych i zasobów.

Aby stworzyć nowy projekt wybieramy:

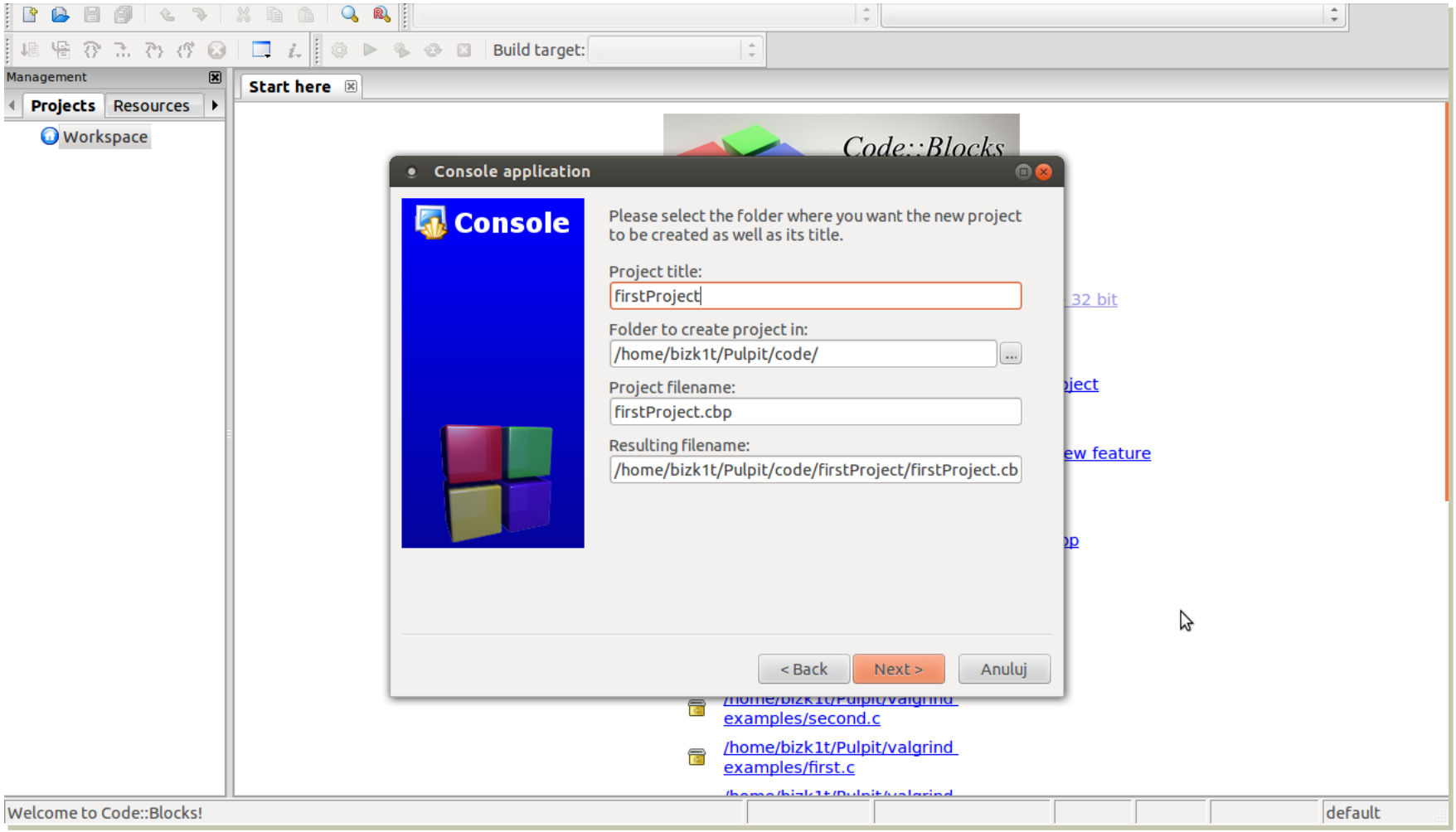
Wybieramy typ projektu.



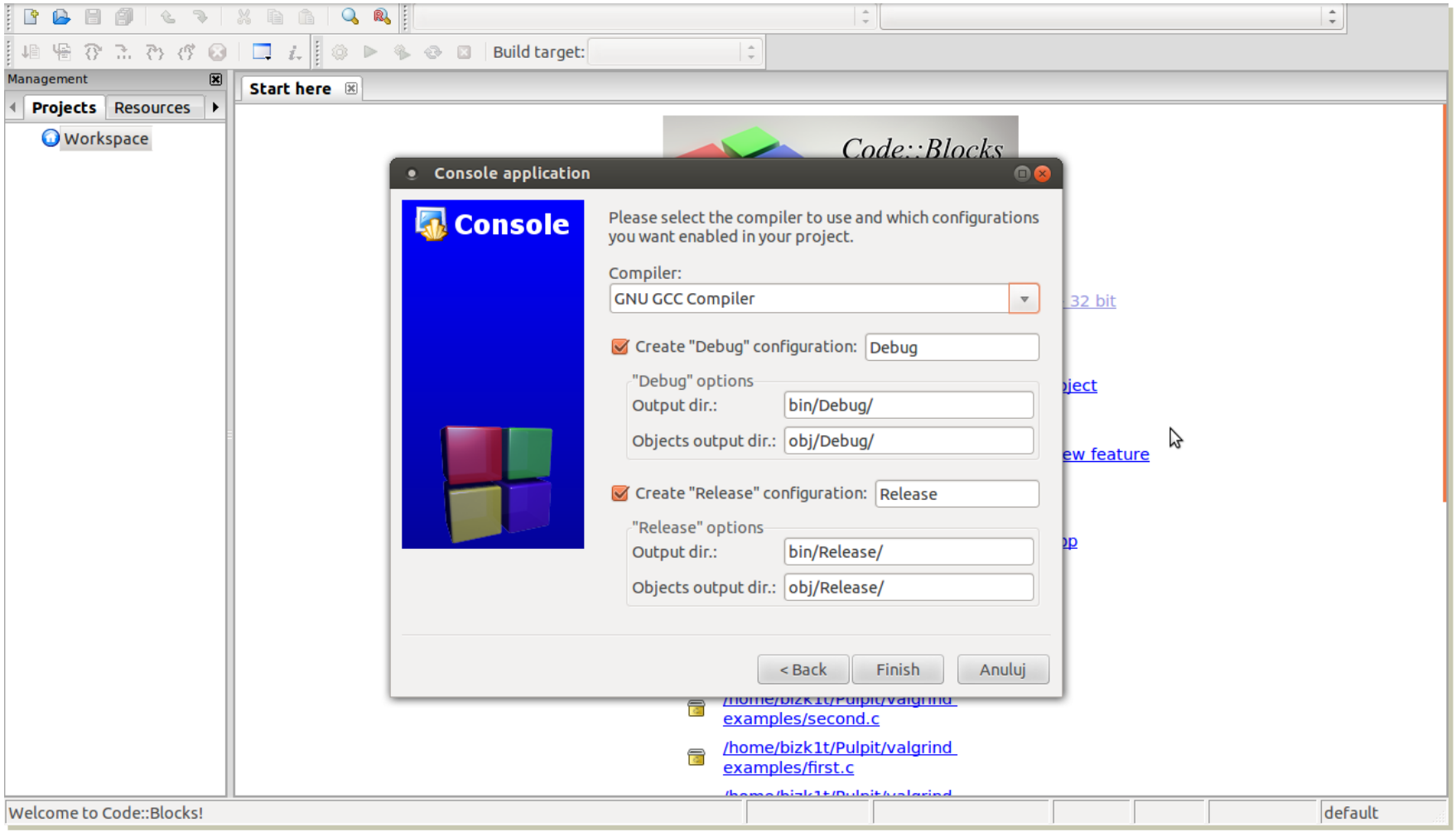
Wybieramy interesujący nas język. (dla projektu konsolowego)



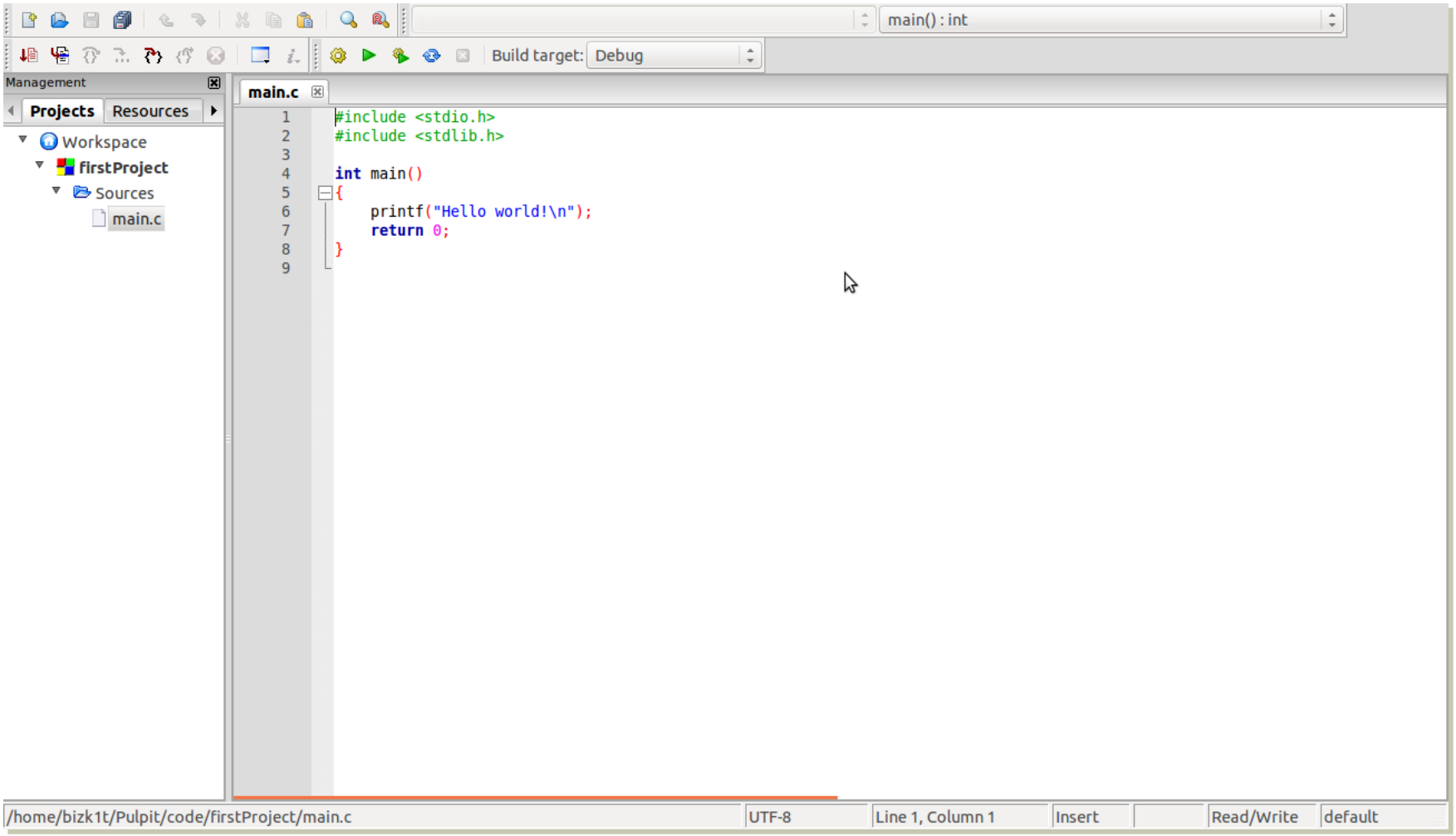
Wpisujemy nazwę projektu oraz ścieżkę docelową.



Wybieramy kompilator oraz opcje związane z wersją debug/release.



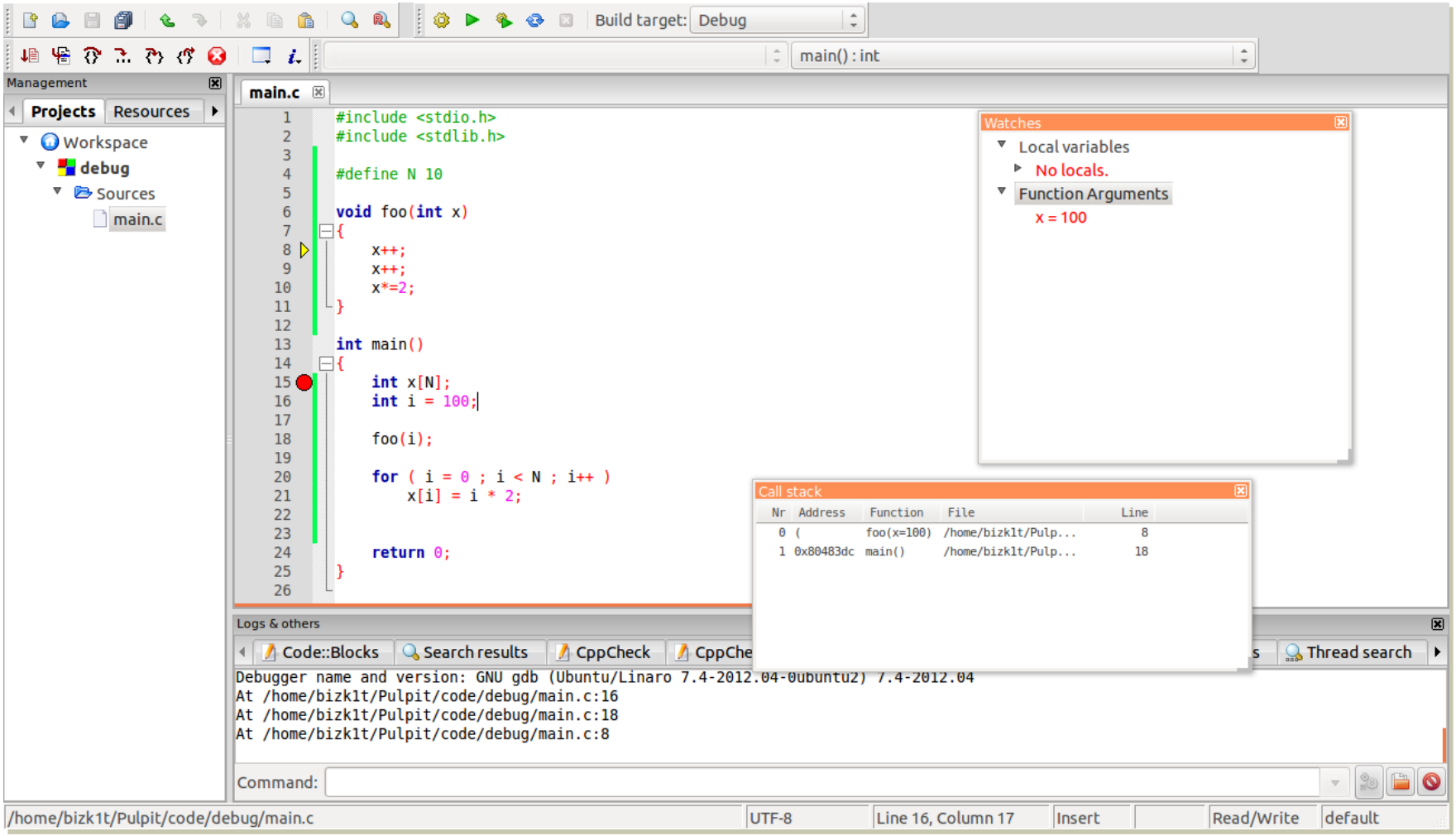
Widok nowego, „surowego” projektu.



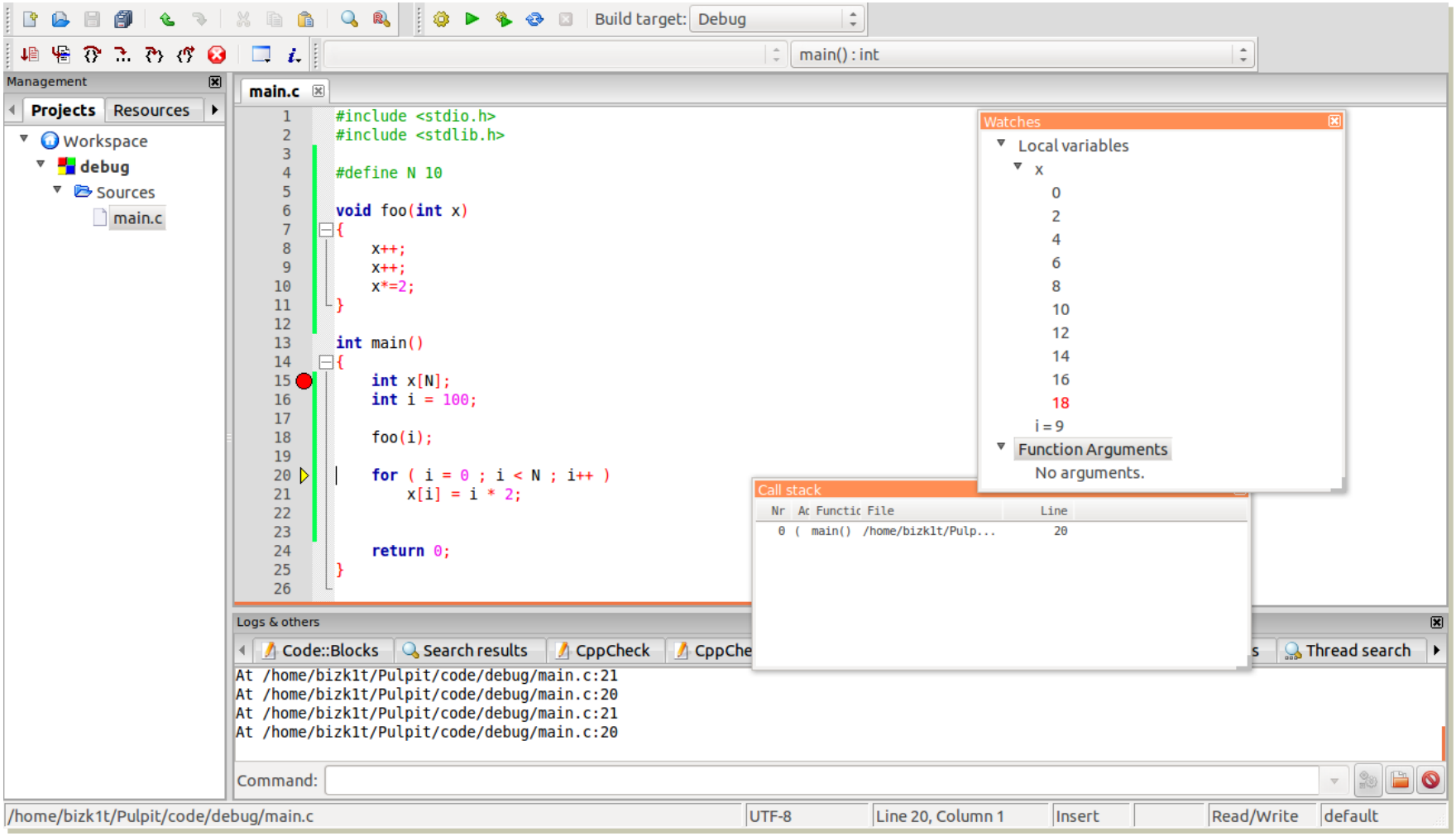
DEBUGGER

Debugger w Code::Blocks spełnia podstawowe funkcjonalności, jednak nie jest tak rozwinięty jak w konkurencyjnych środowiskach (darmowy Eclipse, komercyjny MS Visual Studio), jednak dobrze wykorzystany umożliwia dokładne prześledzenie dowolnej linii w naszym kodzie.

Widok w trakcie debuggowania - włączony breakpoint oraz dwa specjalne okna: podgląd zmiennych i podgląd stosu.



Debuggowanie – ciąg dalszy...



VALGRIND

INSTALACJA KONFIGURACJA

WWW.VALGRIND.ORG

Po ściągnięciu paczki na dysk należy ją rozpakować, następnie wejść do niej oraz wpisać

```
1.  configure
2.  make
3.  make install
```

Po tych czynnościach Valgrind jest gotowy do pracy!

GCC A VALGRIND

Jeśli chcemy korzystać z Valgrinda w pełni, należy zawsze dodać opcję –g do kompilacji. Powoduje to dodanie dodatkowych symboli dla debuggera, dzięki czemu Valgrind potrafi wykryć i podać dokładną linię kodu w której nastąpił wyciek pamięci.

```
gcc source.c –g –o output
```

SKŁADNIA

```
valgrind [ opcje ] ./program [ opcje programu ]
```

przykład

```
valgrind --leak-check=yes --log-file=log.txt ./prog
```

opis

Sprawdź program prog pod kątem wycieków pamięci i zapisz wyniki do pliku log.txt.

Więcej w manualu lub w trakcie prezentacji…

WYKRYWANIE WYCIEKÓW PAMIĘCI

W języku C musimy mieć świadomość, że nie istnieje coś takiego jak Garbage Collector, a więc całe usuwanie niepotrzebnych już zmiennych dynamicznych spoczywa na barkach programisty. Na pomoc przychodzi Valgrind, który potrafi wykryć zmienne niezwolnione z pamięci.

przykład

```
int main()
{
   int* x = malloc( sizeof(int) * 1000 );
   // free( x );
   return 0;
}
```

Tracimy aż

1000 * sizeof(int) bajtów!

Przykład jest dobry dla hipotetycznej sytuacji kiedy system operacyjny nie jest odpowiedzialny za zwalnianie pamięci po skończeniu pracy aplikacji.

WYKRYWANIE ODNIESIEŃ POZA PAMIĘĆ

Ile razy widzieliśmy magiczny napis w trakcie wykonywania programu: „Segmentation fault”? Valgrind pomoże nam znaleźć odniesienie poza alokowaną pamięć.

przykład

```
int main()
{
   int* x = malloc( sizeof(int) * 5);
   x[5] = 123;
   free( x );
   return 0;
}
```

Niby wszystko jest w porządku, ale coś nie gra…

WYKRYWANIE NIEZAINICJALIZOWANYCH ZMIENNYCH

Valgrind potrafi wskazać użycia niezainicjalizowanych zmiennych. Dzięki temu możemy wykluczyć potencjalnie niebezpieczne fragmenty kodu.

przykład

```
int main()
{
   int x;
   if ( x == 5 )
      format_C();
   return 0;
}
```

Raczej nie chcemy, aby pod zmienną x znalazła się przypadkowo wartość 5…

INNE

Valgrind może znaleźć nieprawidłowe użycia funkcji free().

```
int main()
{
   int x;
   free( x );
   return 0;
}
```

Valgrind nie wykryje wyjścia poza zakres tablicy STATYCZNEJ!

```
int main()
{
   int x[5];
   x[8] = 52;
   return 0;
}
```

DZIĘKUJĘ ZA UWAGĘ ☺