



Kurikulum Qt

{ Basic OOP }

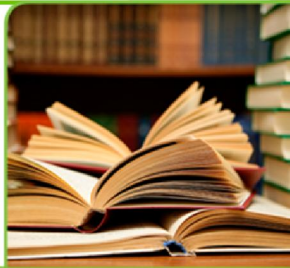
# Chapter 1

Introduction C++  
Development with  
Nokia Qt SDK





# Agenda



- **Pengenalan C dan C++**
- Instalasi Nokia Qt SDK
- Program console pertama
- Fitur-fitur Nokia Qt SDK (QtCreator)
- Struktur Program C++
- Struktur Input dan Output pada C++
- Debugging program dengan QtDebug



# Pengenalan C dan C++

- C++ merupakan pengembangan dari bahasa C
- Bahasa C dikembangkan oleh **Brian W. Kernighan & Dennis M. Ritchie** dari AT & T Laboratories pada tahun 1978
- Sejak tahun 1980 bahasa C mulai digunakan di Eropa
- Tahun 1989, bahasa C distandarkan oleh **American National Standards Institute (ANSI)**.
- Bahasa C yang standar kemudian dikenal dengan nama **ANSI C**.

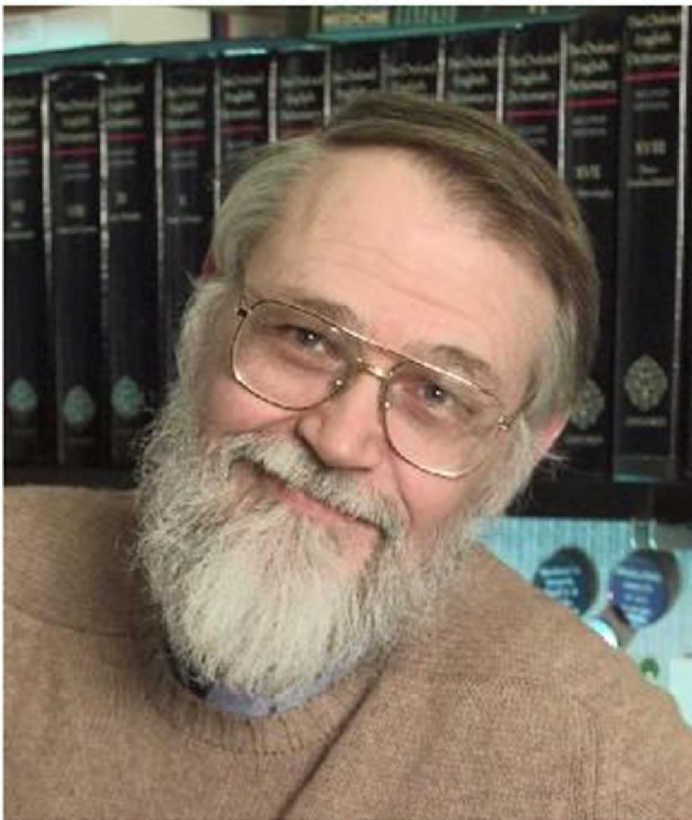


# Pengenalan C dan C++ (2)

- Mulai awal tahun 1980, **Bjarne Stroustrup** dari AT & T Bell Laboratories mulai mengembangkan bahasa C hingga tahun 1985 lahirlah bahasa C++
- Bahasa C++ mengalami dua tahap evolusi:
  - Pertama, dirilis oleh **AT&T Laboratories**, dinamakan **cfront**.
    - C++ versi ini hanya berupa kompiler yang menterjemahkan bahasa C++ menjadi bahasa C untuk dieksekusi
  - Kedua, **Borland International Inc.** mengembangkan kompiler C++ menjadi sebuah kompiler yang mampu mengubah C++ langsung menjadi bahasa mesin (assembly).
    - Tahun 1990, C++ mulai diarahkan ke pengembangan **pemrograman berorientasi obyek**



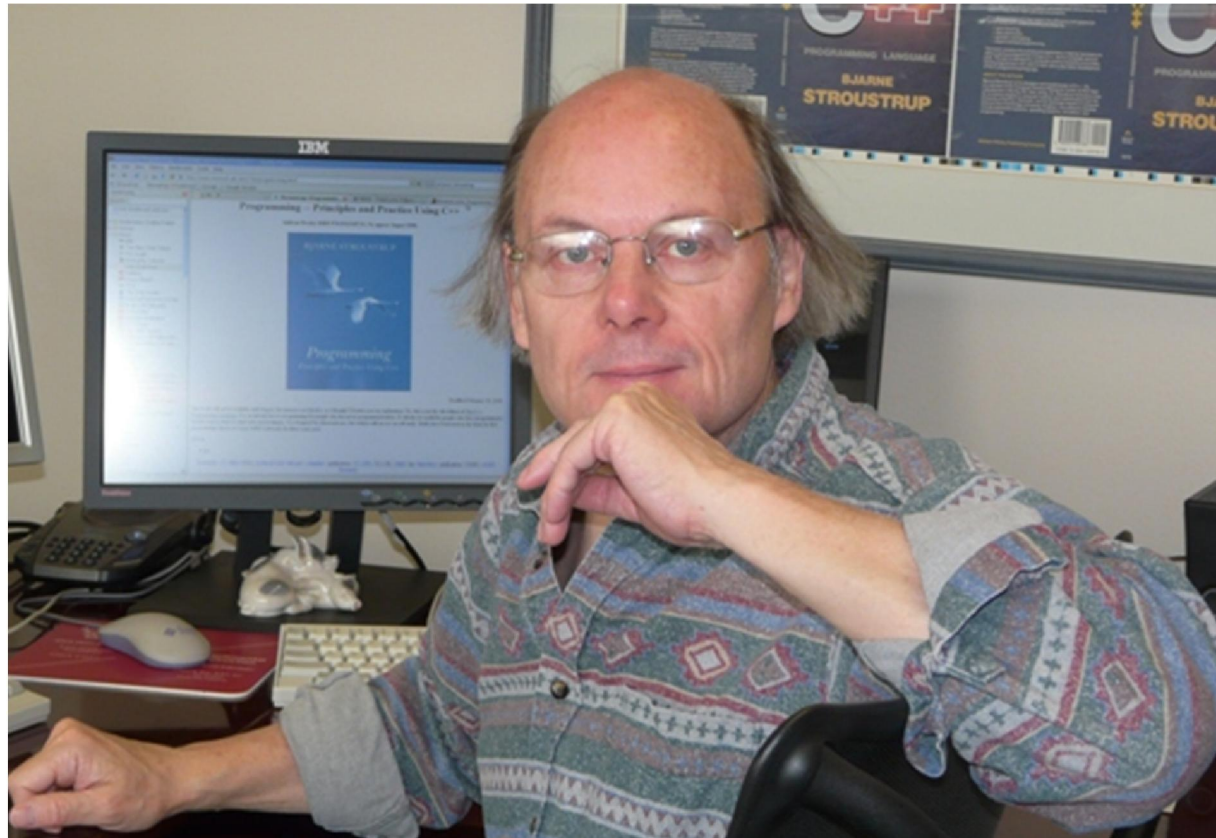
# Penemu bahasa C: **Brian Wilson Kernighan dan Dennis M. Ritchie**







# Penemu bahasa C++: **Bjarne Stroustrup**





# Fitur Bahasa C++

- Berbasis **Berorientasi Obyek**
- **Portable** => dapat digunakan pada berbagai arsitektur komputer
- **Brievity** => kode program ringkas
- Mendukung **Modular Programming** => dengan memisahkan masalah kedalam header file tersendiri yang dapat digunakan kembali (reuse)
- **C Compatible** => backward compatible dengan bahasa C
- **Speed** => output yang dihasilkan cepat dieksekusi pada berbagai jenis arsitektur komputer



# Instalasi Nokia Qt SDK

- SDK = Software Development Kit, suatu alat pengembangan perangkat lunak yang lengkap dan terintegrasi beserta dengan Integrated Development Environment (IDE)-nya
- Tool yang digunakan: Nokia Qt SDK
- Download dari:  
[http://www.forum.nokia.com/info/sw.nokia.com/id/e920da1a-5b18-42df-82c3-907413e525fb/Nokia\\_Qt\\_SDK.htm](http://www.forum.nokia.com/info/sw.nokia.com/id/e920da1a-5b18-42df-82c3-907413e525fb/Nokia_Qt_SDK.htm)
- Nama file:  
**Nokia\_Qt\_SDK\_Win\_offline\_v1\_0\_2\_en.exe**





# Demo Instalasi



# Demo Program Console Pertama



# Fitur-fitur QtCreator

- Advanced C++ code editor
- Project creator wizard.
- Integrated GUI designer.
- Integrated Help (Qt Assistant).
- Visual Studio Add-in and Eclipse Integration
- Cross Platform Build tool
- Version Control System



# Fitur Qt: Advanced C++ Editor

- Code completion
- Find & Replace
- Code Formatting
- Menampilkan baris error dan warning
- Navigating class, function, and symbol
- Provide Context-sensitive Help
- Renaming and Refactoring Support

A screenshot of the Qt C++ editor interface. It shows a C++ code snippet with a dropdown menu for code completion. The code includes `#include <QtCore/QCoreApplication>` and `#include <iostream>`. The `int main` function is defined, and a dropdown menu is open over the `<iostream>` include, showing options: `io.h`, `iomanip`, `ios`, `iosfwd`, and `iostream`. The `iostream` option is highlighted. The code also shows `QCoreApplication a(argc, argv);`, `cout << "Hello World!";`, `co` (part of `cout`), and `return a.exec();`.



# Fitur Qt: **Project Creator Wizard**

- Qt Creator mendukung pembuatan project dengan berbasis wizard sehingga memudahkan kita dapat membuat project secara terpisah dan terstruktur.
- QtCreator C++ project berisi:
  - File-file yang dikelompokkan secara bersama
  - Proses Build yang terkustomisasi secara khusus untuk project tersebut
  - Form dan resource files yang diikutserkan dalam project tersebut
  - Semua setting untuk menjalankan aplikasi dalam project tersebut



# Demo Project Creator Wizard



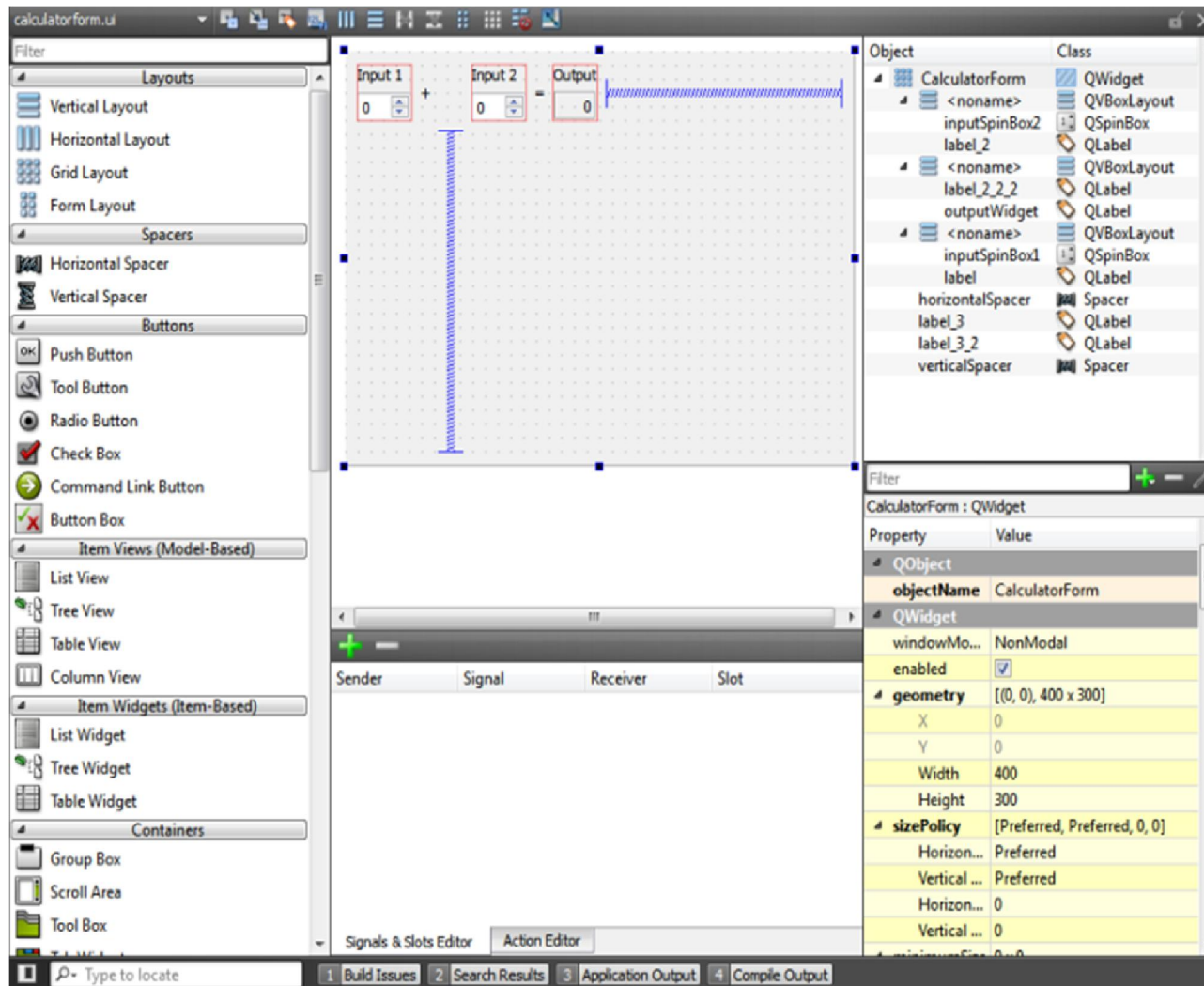


# Fitur Qt: **GUI Builder**

- Desain interface dapat dibuat dengan cepat dengan cepat dengan fasilitas drag and drop
- Dapat melakukan kustomisasi widget atau memilih widget standard yang ada
- Dapat melakukan preview secara real time, dan hasil preview sama dengan yang didesain
- Dapat langsung dihasilkan kode C++ atau Java dari prototipe antarmuka yang dibuat
- Dapat mengintegrasikan Qt Designer dengan Visual Studio atau Eclipse IDE



# GUI Builder





# Visual Studio Add-in and Eclipse Integration

- Menyediakan wizards untuk membuat Qt projects dan classes baru langsung pada VS dan Eclipse
- Dapat secara otomatis build setup untuk Qt Meta-Object Compiler, User Interface Compiler, dan Resource Compiler
- Dapat melakukan import dan export dari Qt Project and Project Include files
- Integrated Qt resource management pada VS dan Eclipse
- Integrated Qt documentation pada VS dan Eclipse
- Debugging extensions for Qt data types pada VS dan Eclipse

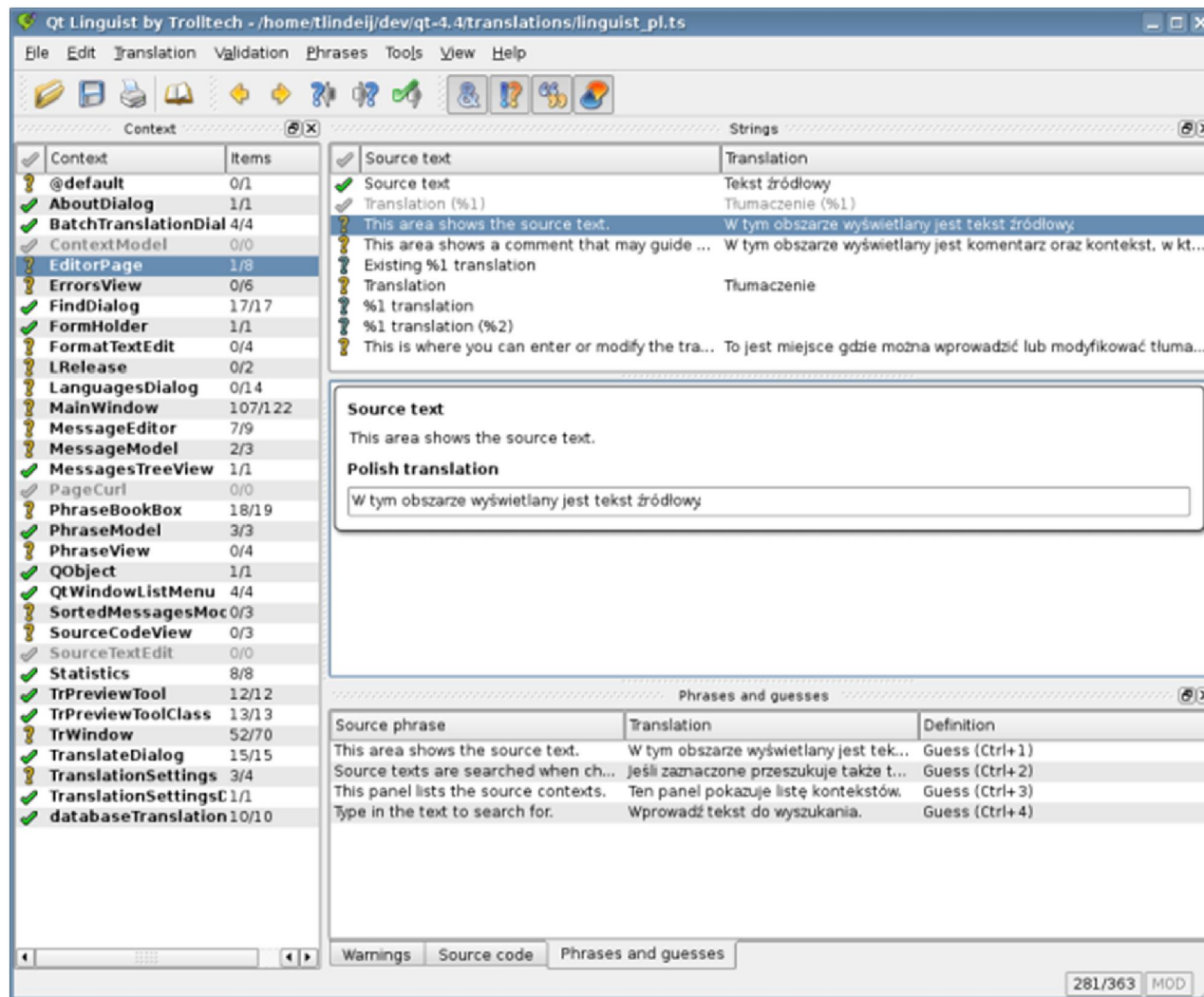


# Fitur Qt: International Translation

- Mengumpulkan dan menyajikan semua teks pada User Interface untuk seorang penerjemah ke dalam sebuah aplikasi sederhana bernama Ling
- Language and font-aware
- Cepat untuk menambahkan bahasa baru untuk aplikasi yang ada dengan alat penggabungan yang cerdas
- Mendukung unicode
- Dapat berpindah-pindah antara bahasa kanan-ke-kiri dan kiri-ke-kanan pada saat runtime
- Dapat mendukung campuran beberapa bahasa dalam satu dokumen aplikasi



# Tampilan Qt Linguist





# Fitur Qt: **Help System**

- Pencarian keyword yang cepat, full text search, indexing dan bookmark pada hasil pencarian
- Kemampuan indexing dan search pada koleksi-koleksi dokumen help secara simultan
- Dokumentasi dapat disimpan secara offline maupun dicari secara online





# Help Systems

```
#include <QtCore/QCoreApplication>
#include <iostream>

int main(int argc, char *argv[])
{
    QCoreApplication a(argc, argv);
    cout<<"Hallo Dunia!";
    cout<<"Percobaan Qt 3.0.0";
    return a.exec();
}
```



## Qt Reference Documentation

Home > Modules > QtCore > QCoreApplication

### QCoreApplication Class Reference

The QCoreApplication class provides an event loop for console Qt applications. [More...](#)

```
#include <QCoreApplication>
```

Inherits QObject.

Inherited by QApplication.

- List of all members, including inherited members
- Qt 3 support members

#### Public Types

#### Contents

- Public Types
- Properties
- Public Functions
- Public Slots
- Signals
- Static Public Members
- Protected Functions
- Related Non-Members
- Macros
- Detailed Description
  - The Event Loop and Handling



# Fitur Qt: Cross Platform Compiler

- Menyederhanakan proses build untuk project pada platform yang berbeda
- Mengotomatiskan proses Makefile generation
- Mempersingkat baris informasi yang diperlukan untuk menciptakan setiap Makefile
- **qmake** juga dapat menghasilkan proyek untuk Microsoft Visual studio tanpa memerlukan perubahan file project



# Fitur Qt: Version Control Systems

- Dapat mendukung berbagai VCS:
  - Git (<http://git-scm.com/>)
  - Subversion (<http://subversion.apache.org/>)
  - Perforce (<http://www.perforce.com/>)
  - CVS (<http://savannah.nongnu.org/projects/cvs>)
  - Mercurial (<http://mercurial.selenic.com/>)
- Fungsi yang tersedia pada QtCreator bergantung pada sistem VCS-nya, meskipun fungsi dasar yang tersedia untuk semua sistem adalah sama
- Fungsi dasar:
  - meng-include-kan file, membandingkan dengan versi terbaru yang tersimpan dalam repositori, dan menampilkan perbedaan, melihat sejarah versioning & rincian perubahan, annotating file, serta melakukan dan , merestore perubahan.

```
Version Control
11:22 Executing: git show --no-color dc62c65
11:25 Executing: git status -u
11:33 Executing: git log --no-color -n 100
11:37 Executing: git log --no-color -n 100 qtcreator.pro
11:38 Executing: git log --no-color -n 100 bin/bin.pro
```



# Fitur Qt: Integrated Debugger

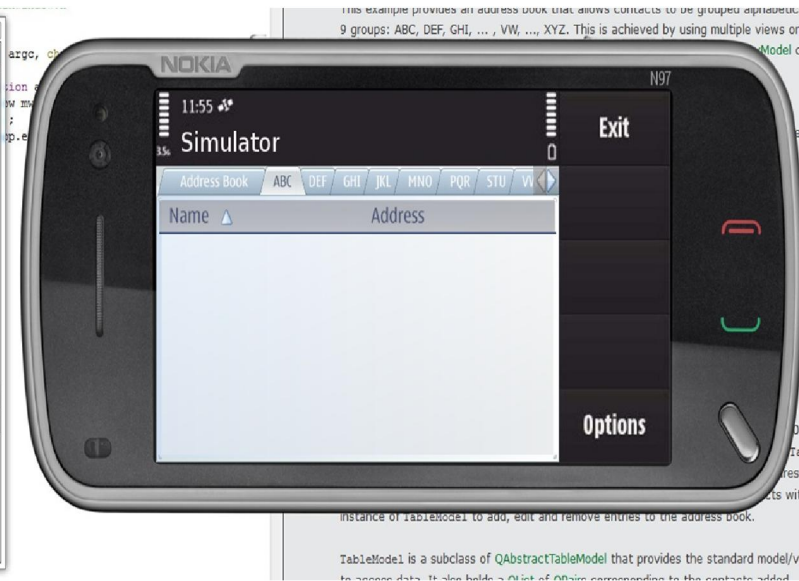
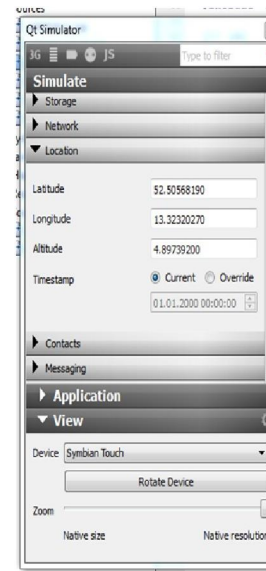
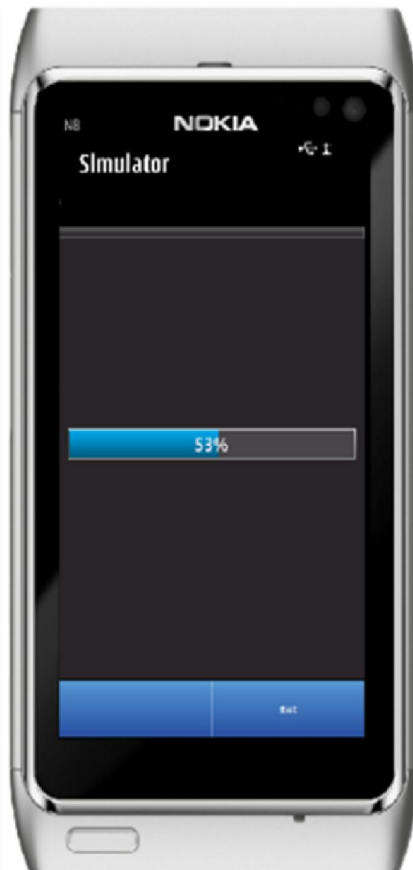
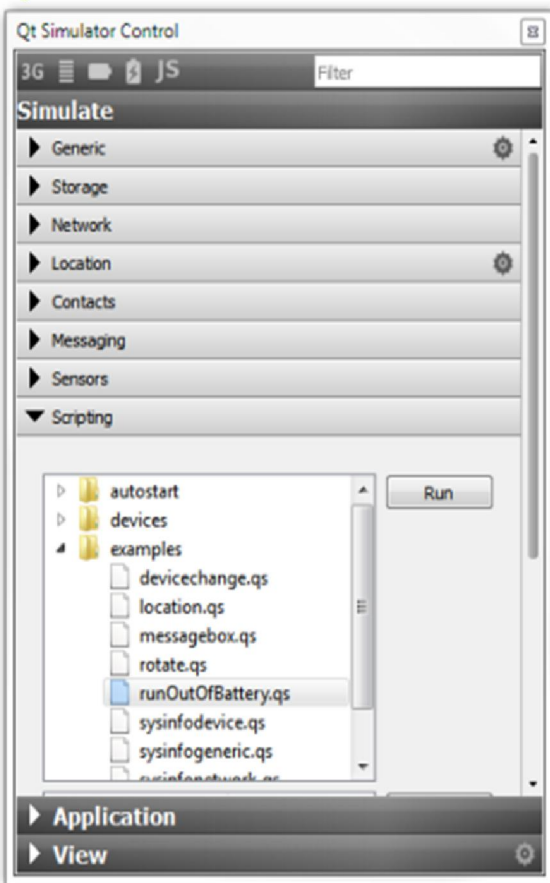
- Mendukung debugger:
  - GNU Symbolic Debugger (**gdb**)
  - Microsoft Console Debugger (**CDB**)
  - Internal Java Script debugger

```
23      QTextStream in(&inputFile);  
24      QString line = in.readAll();  
25      inputFile.close();  
26  
27      ui.textEdit->setPlainText(line);  
28  }  
29
```



# Fitur Qt: Integrated Simulator

- Baru mendukung **Nokia Qt Simulator**





# Qt Creator IDE Interface







# Demo Penggunaan IDE



# Struktur Program C++

- Program Bahasa C/C++ tidak mengenal aturan penulisan di kolom/baris tertentu, jadi bisa dimulai dari kolom/baris manapun.
- Namun demikian, untuk mempermudah pembacaan program dan untuk keperluan dokumentasi, sebaiknya penulisan program di bahasa C/C++ diatur sedemikian rupa sehingga mudah dan enak dibaca.

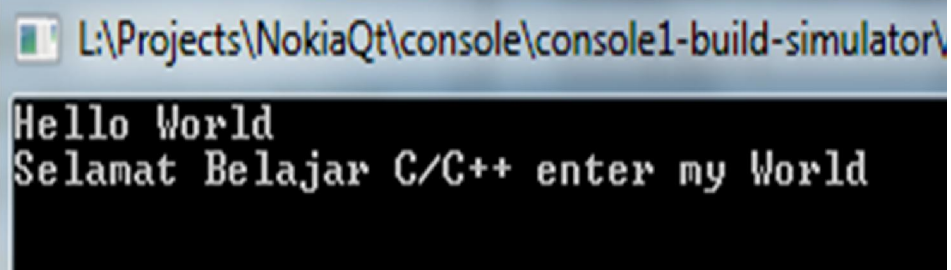


# Struktur Dasar Program C++

```
#include <header>
using namespace std;
int main(int argc, char *argv[])
{
    deklarasi variabel;
    deklarasi konstanta;
    perintah âperintah;
    //komentar
    return 0;
}
```



```
#include <QtCore/QCoreApplication>
#include <iostream>
using namespace std;
int main(int argc, char *argv[])
{
    QCoreApplication a(argc, argv);
    cout<<"Hello World"<<endl;
    cout<<"Selamat Belajar C/C++ ";
    cout<<"enter my World";
    return a.exec();
}
```

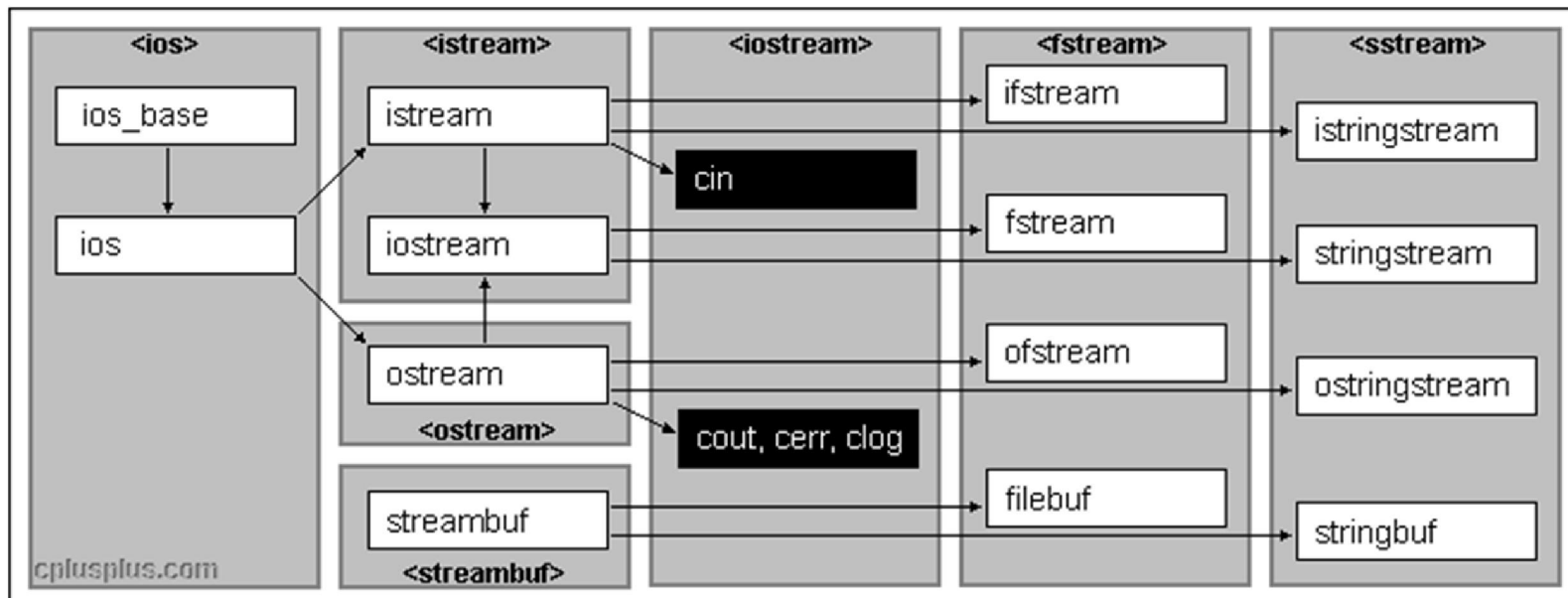


```
L:\Projects\NokiaQt\console\console1-build-simulator\
Hello World
Selamat Belajar C/C++ enter my World
```



# Struktur Input / Output pada C++

- Menggunakan library `<iostream>`
- Menggunakan standard library object stream:
  - `cout<<` , `cerr<<`, dan `clog<<`





# Struktur Output

- Menggunakan standar library **cout**
- Menggunakan konjungsi insertion operator **<<**
- Tanda **<<** dapat digunakan untuk menggabungkan output
- Untuk tanda enter dapat digunakan:
  - Escape character **\n**
  - Fungsi **endl**
- Output dapat diatur tampilannya dengan perintah **setw()** dengan header library **<iomanip>**
- **#include <iomanip>**



# Demo Struktur Output





# Struktur Input

- Menggunakan standar library **cin**
- Menggunakan konjungsi insertion operator **>>**
- cin hanya dapat menerima berbagai jenis data
- Khusus untuk string, secara default cin hanya menerima string hingga ditemukan **blankspace** character
  - Blankspace adalah karakter spasi, tab, enter, backspace
- Untuk menanggulangnya digunakan fungsi **getline()**  
**getline(cin,<identifier>)**



# Demo Struktur Input



# Debugging pada QtCreator

- Debugging adalah kegiatan menelusuri semua tingkah laku, isi data, alur kerja dari program yang dibuat ketika kondisi runtime
- Harapan dari kegiatan debugging adalah kita dapat menemukan kesalahan program yang kita buat jika terdapat kesalahan / error
- QtDebugger:
  - GNU Debugger Simbolik (gdb),
  - Microsoft Console Debugger (CDB),
  - dan Javascript debugger.



# Yang dapat dilakukan oleh QtDebug

- Menuju ke baris program atau instruksi tertentu
- Menginterupsi jalannya program.
- Set breakpoint.
- Memeriksa isi stack pada memory.
- Memeriksa dan memodifikasi register dan isi memori pada saat debugging.
- Memeriksa dan memodifikasi register dan isi memori variabel lokal dan global.
- Memeriksa daftar library bersama yang dibuat.
- Membuat snapshot dari keadaan saat ini ketika program didebug.



# Demo Debugging



# QtDebug

- Merupakan class yang digunakan untuk menampilkan output ke layar
- Menggunakan operator << yang menerima input berupa string
- Harus menggunakan **#include <QtDebug>**
- Contoh:  
**qDebug << "Hallo";**



# Thank You