Effektiv Kode med C og C++

Forelesning 1, vår 2015

Alfred Bratterud

Agenda:

* Litt om meg, og dere
* Om C og C++: historie, relevans, egenskaper
* Praktisk om kurset
* Kode!
* Oppgaver for uka

Om meg:

* Alfred Bratterud, Høgskolelektor og PhD-stipendiat
* alfred.bratterud@hioa.no
* Veldig glad i programmerering:
  * C/C++, Python, javaScript, PHP, Java, Lisp, Perl, Powershell, ActionScript, C#, ASP ...
* Nå mest ifbm. forskning

Litt om dere

* En kjapp quiz av hva dere kan...
  * if, else, for, while, switch, foreach ...
  * public, static, void, main
  * namespace
  * peker
  * generics
  * templates

Litt om dere

* En kjapp quiz av hva dere kan...
  * if, else, for, while, switch, foreach ...
  * public, static, void, main
  * namespace
  * peker
  * generics
  * templates

Må sitte i fingrene,
sammen med variabler, arrays,
funksjoner etc.

Litt om dere

* En kjapp quiz av hva dere kan...
  * if, else, for, while, switch, foreach ...
  * public, static, void, main
  * namespace
  * peker
  * generics
  * templates

Må sitte i fingrene,
sammen med variabler, arrays,
funksjoner etc.

VIKTIG: Dere må like
programmering!

Overordnet plan

* Etter kurset skal man trygt kunne si at man kan C++
  * Det meste av språket
    (keywords, operatorer etc.)
    * En del av standardbiblioteket - for C og STL
    * Kompilering, linking, preprosessering
    * C++11 - mye nytt fra C++98 (forrige store)
* Mest kommandolinje, Noe GUI / Grafikk
  * I prosjektoppgaven velger dere selv

Overordnet plan

* Forstå "effektiv kode" på "høynivå" og "lavnivå"
* Kunne bruke C++ effektivt i et gruppeprosjekt
* All kode skal kjøre på Linux
    * Bruk gjerne windows - men sensurering vil skje i Linux
      (Med mindre særlig god grunn)
* Kompilering, linking og bygging med «Make»

Generell fremgangsmåte

1. Skriv kode i din favoritt-editor
    * Lagre i hello_world.cpp
    * Senere skal vi dele opp i .cpp og .hpp (eller .h)
2. Kompilér i terminal med GCC (g++)
    * g++ hello_world.cpp -o hello_world
    * Senere skal vi automatisere med GNU Make
3. Kjør fra terminal
    * Finn mappen (cd ~/my/cpp/folder)
    * ./hello_world
4. Gjenta fra steg 1 til du er fornøyd

Hvordan ser koden ut?

hello_world.cpp

Hvordan ser koden ut?

hello_world.cpp

```cpp
#include <iostream>
using namespace std;

int main(){
  cout << "Hello World!" << endl;
}
```

Hello world...

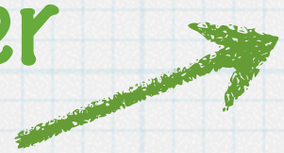
Hvordan ser koden ut?

hello_world.cpp

```cpp
#include <iostream>
using namespace std;

int main(){
  cout << "Hello World!" << endl;
}
```

```cpp
void simple_loop(){

  for(int i = 0; i < 100; i++)
    cout << "Hello " << i << endl;

}
```

Funksjoner og løkker «som vanlig»



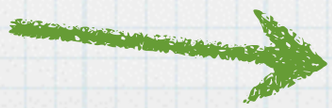
Hvordan ser koden ut?

hello_world.cpp

```cpp
#include <iostream>
using namespace std;

int main(){
  cout << "Hello World!" << endl;
}
```

```cpp
void simple_loop(){

  for(int i = 0; i < 100; i++)
    cout << "Hello " << i << endl;

}
```

Matematikk som forventet



```cpp
float math_as_expected(float x){
  return 5*pow(x,2) + 17.2*cos(x) + 10.9;
}
```

```
IP_stack() :
  _eth(Ethernet::addr({0x08,0x00,0x27,0x9D,0x86,0xE8})),
  _arp(Ethernet::addr({0x08,0x00,0x27,0x9D,0x86,0xE8}),
       IP4::addr({(uint8_t)192,(uint8_t)168,(uint8_t)0,(uint8_t)11})),
  _ip4(IP4::addr({(uint8_t)192,(uint8_t)168,(uint8_t)0,(uint8_t)11}))
{

  printf("<IP Stack> constructing \n");

  /** Make delegates for bottom of layers */
  auto arp_bottom(delegate<int(uint8_t*,int)>::from<Arp,&Arp::bottom>(_arp));
  auto ip4_bottom(delegate<int(uint8_t*,int)>::from<IP4,&IP4::bottom>(_ip4));
  auto ip6_bottom(delegate<int(uint8_t*,int)>::from<IP6,&IP6::bottom>(_ip6));


  auto icmp_bottom(delegate<int(uint8_t*,int)>::from<ICMP,&ICMP::bottom>(_icmp));
  auto udp_bottom(delegate<int(uint8_t*,int)>::from<UDP,&UDP::bottom>(_udp));

  // Hook up layers on top of ethernet
  // Upstream:
  _eth.set_arp_handler(arp_bottom);
  _eth.set_ip4_handler(ip4_bottom);
  _eth.set_ip6_handler(ip6_bottom);

  _ip4.set_icmp_handler(icmp_bottom);
  _ip4.set_udp_handler(udp_bottom);


}
```

Hvo r?

Hrrrmm...

```
#include <iostream>
using namespace std;

int main(){
  cout << "Hello World!" << endl;
}
```

```
void simple_loop(){

  for(int i = 0; i < 100; i++)
    cout << "Hello " << i << endl;

}
```

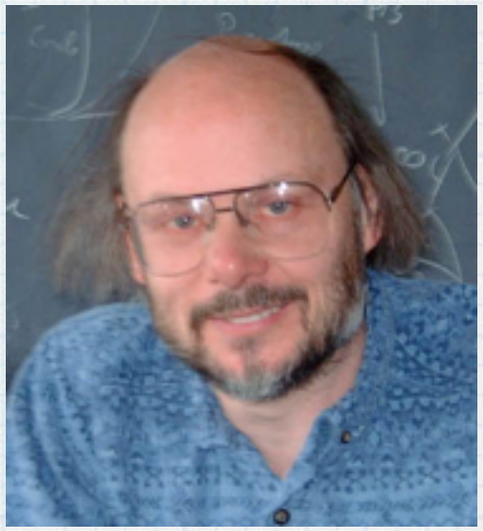
```
float math_as_expected(float x){
  return 5*pow(x,2) + 17.2*cos(x) + 10.9;
}
```

Litt historie

* C ble utviklet av Dennis Ritchie v. Bell Labs, og kom ut i 1972

Dennis Ritchie, 1941-2011

* C++ ble utviklet av Bjarne Stroustrup, først kalt også ved Bell labs. C++ kom ut i 1983



Bjarne Stroustrup, f. 1950

* Het opprinnelig «C with classes»

* Objektorientert programmering ble oppfunnet av to nordmenn (Yay!) Ole Johan Dahl og Kristen Nygaard, allerede i 1967

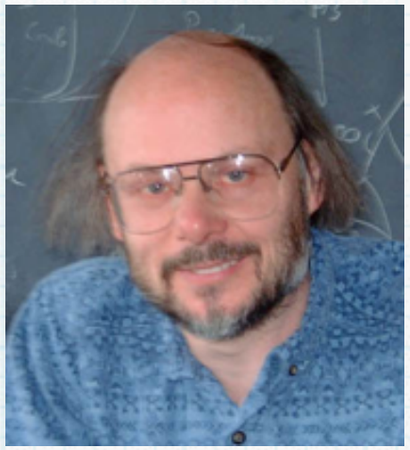
Kristen Nygaard

Ole Johan Dahl

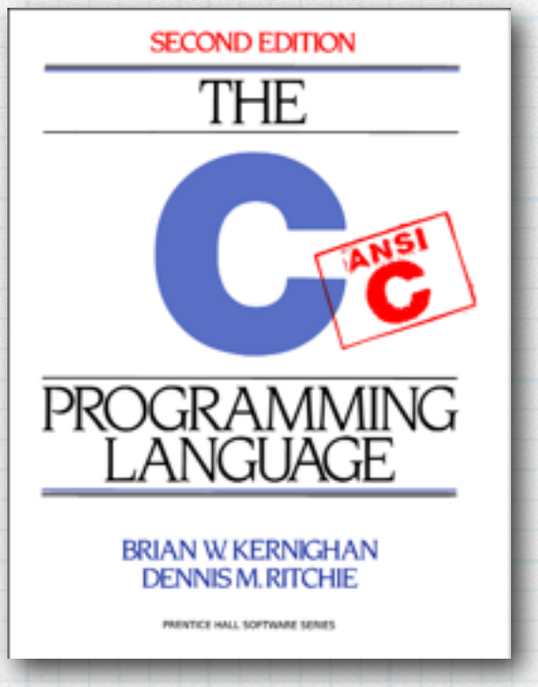
To viktige - og bra bøker

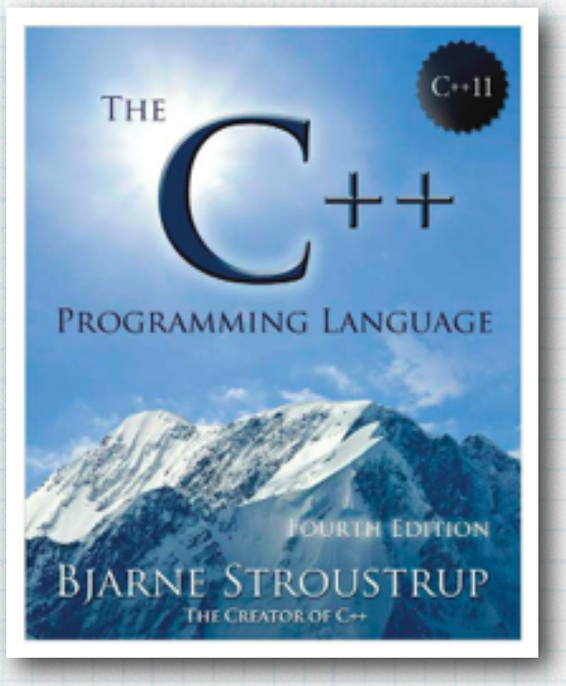
C++ - boken er støttelitteratur i faget

Dennis Ritchie



Biarne Stroustrup





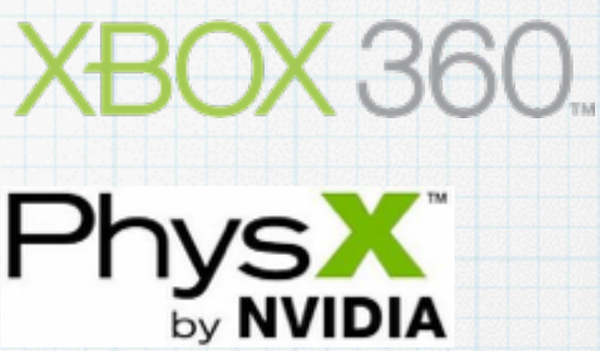
Hva brukes C++ til i 2015?


Linux


Windows 8


OS X Yosemite
Every bit as powerful as it looks.

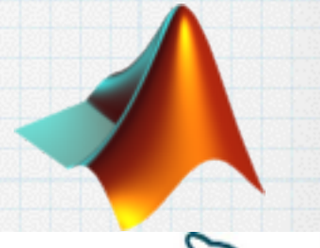

WORLD OF WARCRAFT


XBOX 360


CALL OF DUTY
ADVANCED WARFARE
RP


grand theft auto V

PhysX
by NVIDIA




MySQL


python


ADOBE PHOTOSHOP 6


Autodesk
Maya


Houdini


Spotify


Qt

“you must know windows and C++ programming”

http://msdn.microsoft.com/en-us/library/windows/apps/hh452744%28v=VS.85%29.aspx

“Developers generally create games using C++ and DirectX 11.1”

http://www.xbox.com/en-US/developers/faq

Hvorfor bruker de C++ ?

* Kort oppsummert: Effektivitet
* Hvorfor så effektivt?
  * Direkte kompilert for hardware
  * Lavnivå minnetilgang
  * Avansert objektorientering
  * Kan "Alt"

Språklige forskjeller

(Ikke fullstendig, ikke representativt utvalg)

Egenskap	C	C++	Java	PHP
Imperative	x	x	x	x
Structured	x	x	x	x
Static types	x	x	x	
Type safe by default		(x)	x	-
OS-uavhengig	(x)	(x)	x	x
Generics		x	x	-
Object Oriented		x	x	x
Friendships		x		
Multiple Inheritance		x		
Functional	(x)	(x)		
Low-level memory	x	x		
Natively compiled	x	x		
Inline Assembly	x	x		
Operator overloading		x		
Garbage Collection			x	x

Så C++ er best i alt?

* Vel...
  * C++ er vanskeligere (dyrere) å bli god i
  * Det er lettere å gjøre alvorlige feil
  * ...Som java, men med "All the guns and all the knives"
  * C++ følger "Zero Overhead principle": Ingen ting som går utover ytelsen blir tatt med uten at du ber om det (eller lager det)
  * Man har altså ingen hushjelp (Garbage collector)
  * Men man kan lære å lage 0 søppel og ikke å sløse.

Blir det C eller C++ ?

* C ⊂ C++ (nesten*)
* Vi skal lære C++
* ...Men bli kjent med mange av de tingene som kommer fra C
* Er du i tvil, gjør det på C++ -måten

* http://en.wikipedia.org/wiki/Compatibility_of_C_and_C%2B%2B

Praktisk

Tid og sted

* Forelesning:
    * Tirsdag kl. 10:30
* Lab:
    * Tirsdag kl. 12:30 - 16:15
    * Jeg vil ofte bli med i starten (Legger inn rom for lunsj)
    * Går gjerne gjennom en del praktiske eksempler / ting mange sliter med
    * Stud.ass fra 12:30

Vurdering

* 2 obliger (arbeidskrav)
* 1 individuell prøve
* 1 prosjekt i gruppe
* Presentasjon av prosjektet for meg
* Mappeevaluering (A-F)
* Detaljert plan kommer!

Github

* Alt blir lagt ut på github: https://github.com/hioa-cs/cpp_v2015
* "Watch" repoet for å få varsler om oppdateringer!

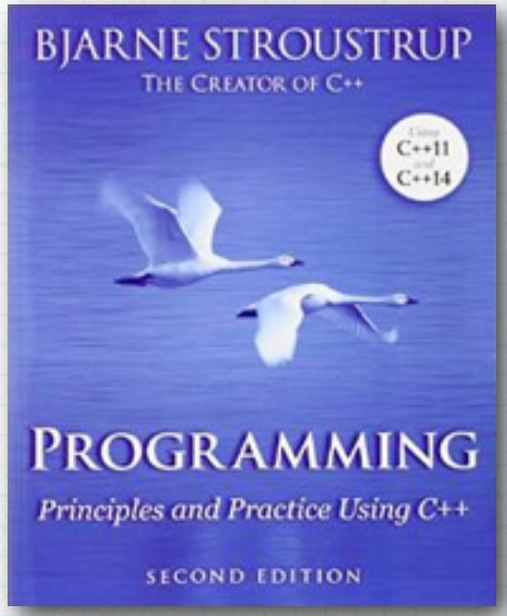
* Alle innleveringer skjer *også* gjennom github.
* Alle må ha studentkonto på github, registrert med sin skole-e-post: https://github.com/edu
* HiOA-adresser skal være på whitelist
* Når du har registrert deg, lever link til din github-bruker på fronter. **Obligatorisk!**

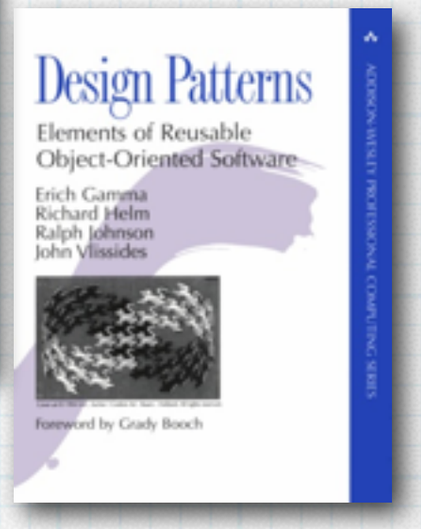
Oppsett av system

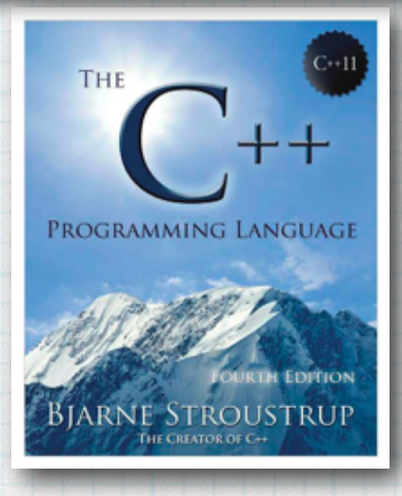
* Vi trenger:
  * Kompilator (GCC) med linker (følger med)
  * Enkel Tekst-editor med syntaksmerking
    * Jeg bruker emacs - men feks. sublimeText helt OK
    * Ikke bruk «store» IDE'er som NetBeans / VS - de gjør for mye for dere
      (Og visualStudio bruker ikke GCC)
* Anbefaler å jobbe på egen PC
  * Men i virtuell maskin - Linux i VirtualBox
    (Legges ut i lab)
* Sett gjerne opp lokalt - MEN: Alt skal kjøre på Linux
* Sysadmin er også gøy... men gir ingen karakter :-)

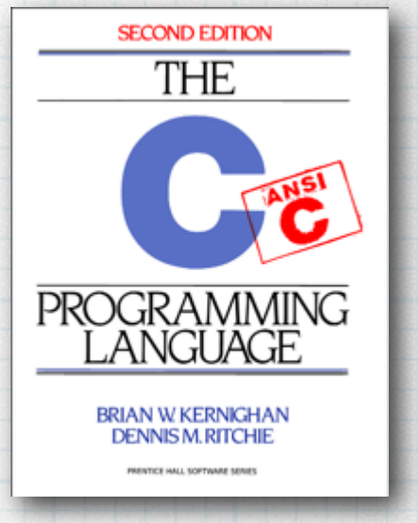
Pensum

- "Programming: Principles and Practices Using C++", B. Stroustrup
- Web: cplusplus.com tutorial
- Støttelitteratur
  - Web: C++ Annotations
  - Gamma, Helm ++ "Design Patterns"
  - Stroustrup, "The C++ Prog.Lang."
  - Ritchie, "The C Prog. Lang."
  - C++ GUI Programming with Qt4









Oppgave: Roulette

* Hvis man da ser det har kommet 9 like på rad - bør man ikke da stase stort på motsatt farge?
* Ved et "fair" roulette-bord skal det være 50/50 for sort og rød
* Tilfeldige variabler er normalfordelt
* Altså bør det veldig sjelden komme seks like på rad, og ekstremt sjelden 10 på rad...?
* Kan løses med enkel statistikk - men hvorfor ikke bare simulere det?

Oppgave: Roulette

* Jeg har laget en Python-implementasjon på github
* I dag: Forsøk å implementere det samme programmet i dit favorittspråk
  * Deretter i C++
* Hva hvis jeg ikke liker/kan matte!?
  * Matte er ikke fokus i kurset - og du trenger ikke skjønne den.
  * Men - du bør kunne implementere en formel

Roulette

roulette.cpp - Vi starter her, fortsetter i lab

TODO Lab:

* Få til «Hello World» I virtualBox
    * Evt. lokalt om du vil
* Opprett *privat* kurs-repo på github
    * Du må registrere en studentkonto med din HiOA e-post
    * Når OK - levér link til repo i fronter
* Skriv roulette-kode i ditt favorittspråk
* Skriv roulette-kode i C++