Metodi Computazionali della Fisica
Secondo Modulo: C++

Prima Lezione

Presentazione del corso

Introduzione a subversion

Input/Output

Implementazione di un generatore di dati artificali

Introduzione a doxygen

In breve

Obiettivi:

- imparare a sviluppare e gestire un progetto di programmazione;
- conoscere tenciche di analisi numerica;
- sviluppare un codice funzionante per l'analisi di dati.

Contenuti, orario e risorse:
`http://www.ge.infn.it/~piccione/metodicomputazionali/`

Creazione della repository:

```
> svnadmin create --fs-type fsfs
     /home/piccione/metodi_computazionali_2/repository/

-rw-r--r-- 1 piccione users  229 Nov 29 12:20 README.txt
drwxr-xr-x 2 piccione users 4096 Nov 29 12:20 conf
drwxr-xr-x 2 piccione users 4096 Nov 29 12:20 dav
drwxr-sr-x 5 piccione users 4096 Nov 29 12:20 db
-r--r--r-- 1 piccione users    2 Nov 29 12:20 format
drwxr-xr-x 2 piccione users 4096 Nov 29 12:20 hooks
drwxr-xr-x 2 piccione users 4096 Nov 29 12:20 locks
```

Definizione della struttura del progetto:

```
> mkdir mc2_cpp
> cd mc2_cpp/
> mkdir src
> mkdir doc
> mkdir devel
> cd src/
```

Creazione di file di esempio:

```
> echo "first main" > main.cpp
> more main.cpp
first main
> cd src/
> echo "first module" > module.cpp
> cd ../doc/
> echo "first document" > document.cpp
> cd ../devel/
> echo "first devel" > devel.cpp
cd ../..
```

Prima sottomissione:

```
> svn import mc2_cpp \
  file:///home/piccione/metodi_computazionali_2/repository/ \
     -m "Initial import"
Adding         mc2_cpp/devel
Adding         mc2_cpp/devel/devel.cpp
Adding         mc2_cpp/doc
Adding         mc2_cpp/doc/document.cpp
Adding         mc2_cpp/main.cpp
Adding         mc2_cpp/src
Adding         mc2_cpp/src/module.cpp

Committed revision 1.

rm -rf mc2_cpp
```

Checkout:

```
> svn checkout  \
  file:///home/piccione/metodi_computazionali_2/repository/ prova
A    prova/devel
A    prova/devel/devel.cpp
A    prova/doc
A    prova/doc/document.cpp
A    prova/main.cpp
A    prova/src
A    prova/src/module.cpp
Checked out revision 1.
```

Info e log:

```
> cd prova/
> svn info
Path: .
URL: file:///home/piccione/metodi_computazionali_2/repository
Repository Root: file:///home/piccione/metodi_computazionali_2/repository
Repository UUID: 97f55880-0f40-0410-9234-fe62ef0d3322
Revision: 1
Node Kind: directory
Schedule: normal
Last Changed Author: piccione
Last Changed Rev: 1
Last Changed Date: 2007-11-29 12:34:09 +0100 (Thu, 29 Nov 2007)

> svn log
------------------------------------------------------------------------
r1 | piccione | 2007-11-29 12:34:09 +0100 (Thu, 29 Nov 2007) | 1 line

Initial import
------------------------------------------------------------------------
```

Add, move, restore:

```
> svn move src modules
A         modules
D         src/module.cpp
D         src
> cd modules/
> ls
total 4
-rw-r--r-- 1 piccione users 13 Nov 29 12:39 module.cpp

> echo "first header" > header.cpp
> svn add header.hpp
A         header.hpp

> touch toberemoved.txt
> svn add toberemoved.txt
A         toberemoved.txt

> cd ../doc
>  rm document.cpp
rm: remove regular file ‘document.cpp’? y
> ls
> svn update
Restored 'document.cpp'
At revision 2.
```

Seconda sottomissione e checkout:

```
> svn commit -m "Second submission"
Adding         modules
Adding         modules/header.hpp
Adding         modules/toberemoved.txt
Deleting       src
Transmitting file data ..
Committed revision 2.
cd ..
rm -rf prova

> svn checkout  \
   file:///home/piccione/metodi_computazionali_2/repository/ svn2
A    svn2/devel
A    svn2/devel/devel.cpp
A    svn2/doc
A    svn2/doc/document.cpp
A    svn2/main.cpp
A    svn2/modules
A    svn2/modules/module.cpp
A    svn2/modules/toberemoved.txt
A    svn2/modules/header.hpp
Checked out revision 2.
```

Status e editor:

```
A Resource is scheduled for Addition
D Resource is scheduled for Deletion
M Resource has local Modifications
C Resource has Conflicts (changes have not been completely merged
    between the repository and working copy version)
X Resource is eXternal to this working copy (may come from another
    repository). See the section called Externals Definitions
? Resource is not under version control
! Resource is missing or incomplete (removed by another tool than
    Subversion)

> export SVN_EDITOR='emacs -nw'
```

Esportare la repository:

```
> svnadmin dump \
  /home/piccione/metodi_computazionali_2/repository > old_rep.dump
* Dumped revision 0.
* Dumped revision 1.
* Dumped revision 2.
```

Importare la repository:

```
> svnadmin create repository
> svnadmin load repository < old_rep.dump
<<< Started new transaction, based on original revision 1
     * adding path : devel ... done.
     * adding path : devel/devel.cpp ... done.
     * adding path : doc ... done.
     * adding path : doc/document.cpp ... done.
     * adding path : main.cpp ... done.
     * adding path : src ... done.
     * adding path : src/module.cpp ... done.

------- Committed revision 1 >>>

<<< Started new transaction, based on original revision 2
     * adding path : modules ...COPIED... done.
     * adding path : modules/header.hpp ... done.
     * adding path : modules/toberemoved.txt ... done.
     * deleting path : src ... done.

------- Committed revision 2 >>>
```

Repository remota:

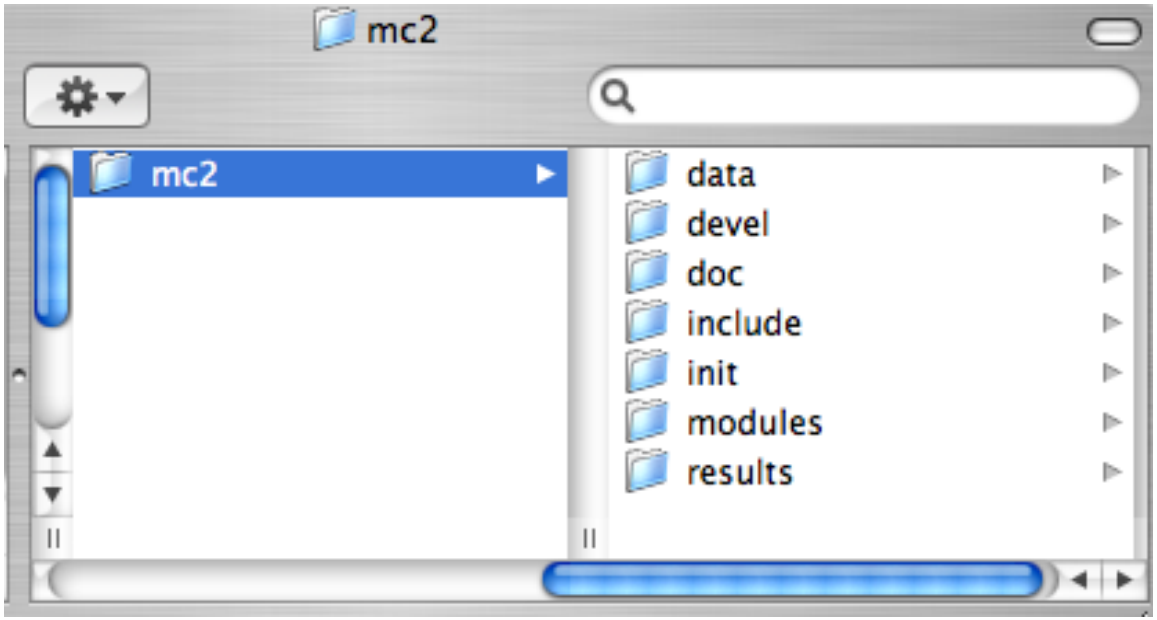
```
somewhere> emacs conf/svnserve.conf -nw

[general]
anon-access = none
auth-access = write
authz-db = authz
[tunnels]
ssh = ssh

somewhere> svnserve -d

here> svn co \
 svn+ssh://piccione@somewhere.it/home/piccione/tmp/repository/ prova
```

Riferimenti

- `http://subversion.tigris.org/`;
- SVN book, Appendix A.

Esercizio

Realizzate una struttura di directory tipo quella in figura e fate un commit nella repository.



Leggere e scrivere su un file:

```
echo "1 2 5 6" > prova.in

--------------------------------------
#include <fstream>

using namespace std;

int main(void) {

  int temp;
  ifstream fin( "prova.in" );
  ofstream fout( "prova.out");
  while( fin >> temp ){
    fout << temp + 2 << endl;
  }
  fin.close();
  fout.close();
  return 0;
}
```

Append:

```
#include <fstream>

using namespace std;

int main(void) {

  int temp;
  ifstream fin( "prova.in" );
  ofstream fout( "prova.out",ios::app);
  while( fin >> temp ){
    fout << temp + 2 << endl;
  }
  fin.close();
  fout.close();
  return 0;
}
```

Trovate le diverse opzioni di `ios::` in *Thinking in C++, vol. 2*, pag. 76.

Specificare i nomi dei file:

```
#include <fstream>
#include <iostream>

using namespace std;

int main(void) {

  string file_in;
  string file_out;

  cin>>file_in>>file_out;

  int temp;
  ifstream fin;
  ofstream fout;

  fin.open(file_in.data());
  fout.open(file_out.data());

  if( !fin ) {
    cerr << "Error: file "<<file_in<<" doesn't exist" << endl;
    return 1;
  }

  while( fin >> temp ){
    fout << temp + 2 << endl;
  }
  fin.close();
  fout.close();
  return 0;
}
```

STL - fstream

```
#include <fstream>
  ifstream( const char *filename, openmode mode );
  ofstream( const char *filename, openmode mode );
  void open( const char *filename );
  void open( const char *filename, openmode mode = default_mode );
```

Nota Bene: il metodo data applicato ad una stringa restituisce un puntatore a carattere.

Premessa

- Per poter testare un codice di analisi dati dobbiamo essere sicuri che funzioni su dati conosciuti;
- vogliamo analizzare dati con errore (misure sperimentali), quindi anche i dati artificiali dovranno avere un errore (rumore);
- possiamo generare questo errore con un generatore di numeri casuali (ad esempio, il rumore termico della CPU);
- meglio usare un generatore *pseudo* casuale in modo tale che i numeri generati siano riproducibili;
- visto che vogliamo simulare dei dati sperimentali, prendiamo i numeri distribuiti secondo una distribuzione gaussiana unitaria centrata nell'origine.

Un esempio di generatore pseudo-casuale

$$I_{j+1} = aI_j + c(\mathrm{mod}m)\,,$$

dove tutti i numeri sono interi e

$$b \equiv c(\mathrm{mod}m) \leftrightarrow \frac{b-c}{m} \in \mathbb{Z}\,.$$

In C++ mod è l'operatore %. Possibili valori dei coefficienti:

a	c	m
106	1283	6075
211	1663	7875
65539	0	2^{31}

Status e editor:

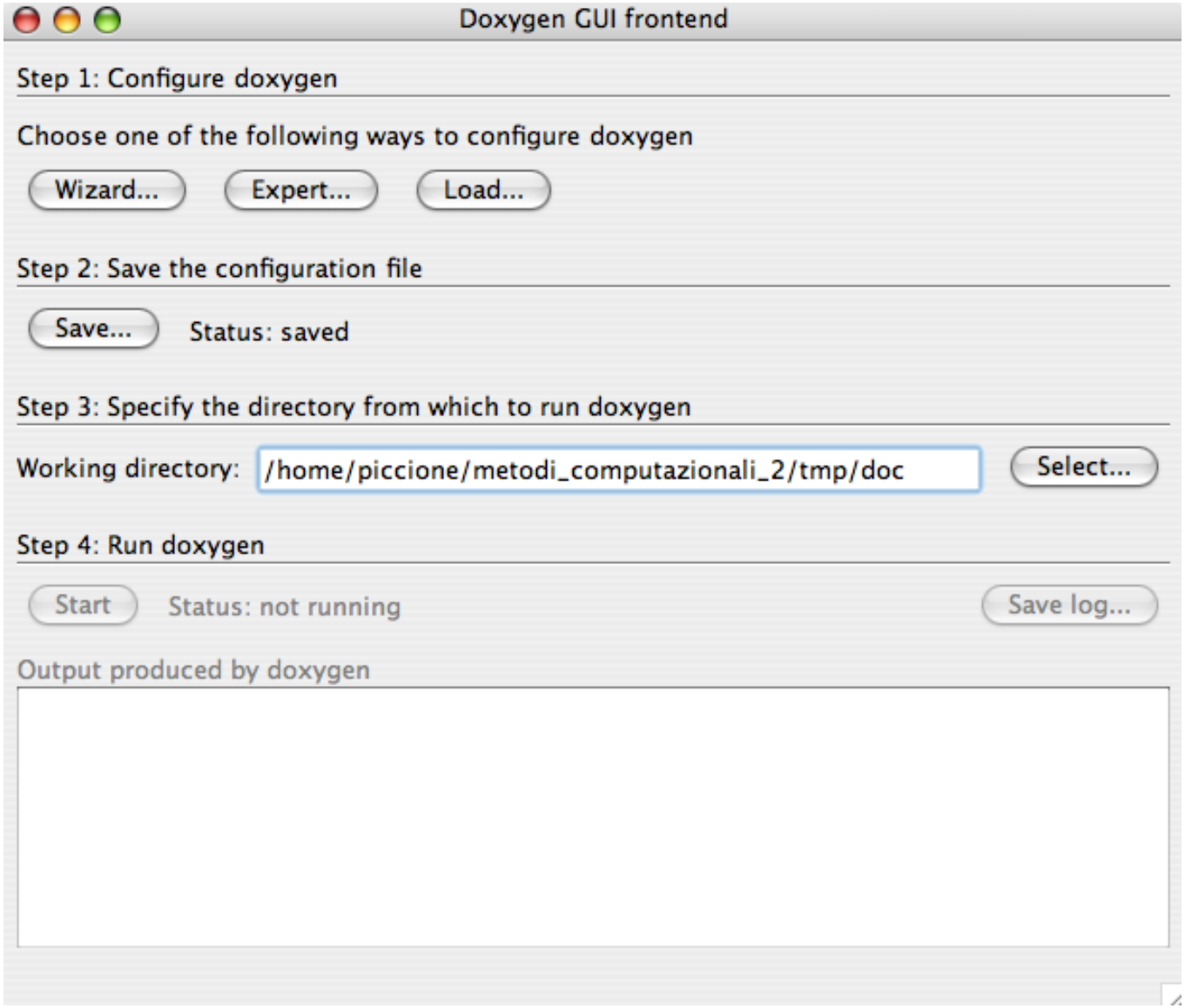
1. Copiate i file `ran.cpp` e `ran.hpp` dalla directory: `/home/piccione/metodi_computazionali_2/riferimenti/`;
2. nella vostra directory `devel` create un programma che generi dei dati con le funzioni

$$y = x^2\,, \qquad y = 0.5 + 0.4\sin(2\pi x)\,,$$

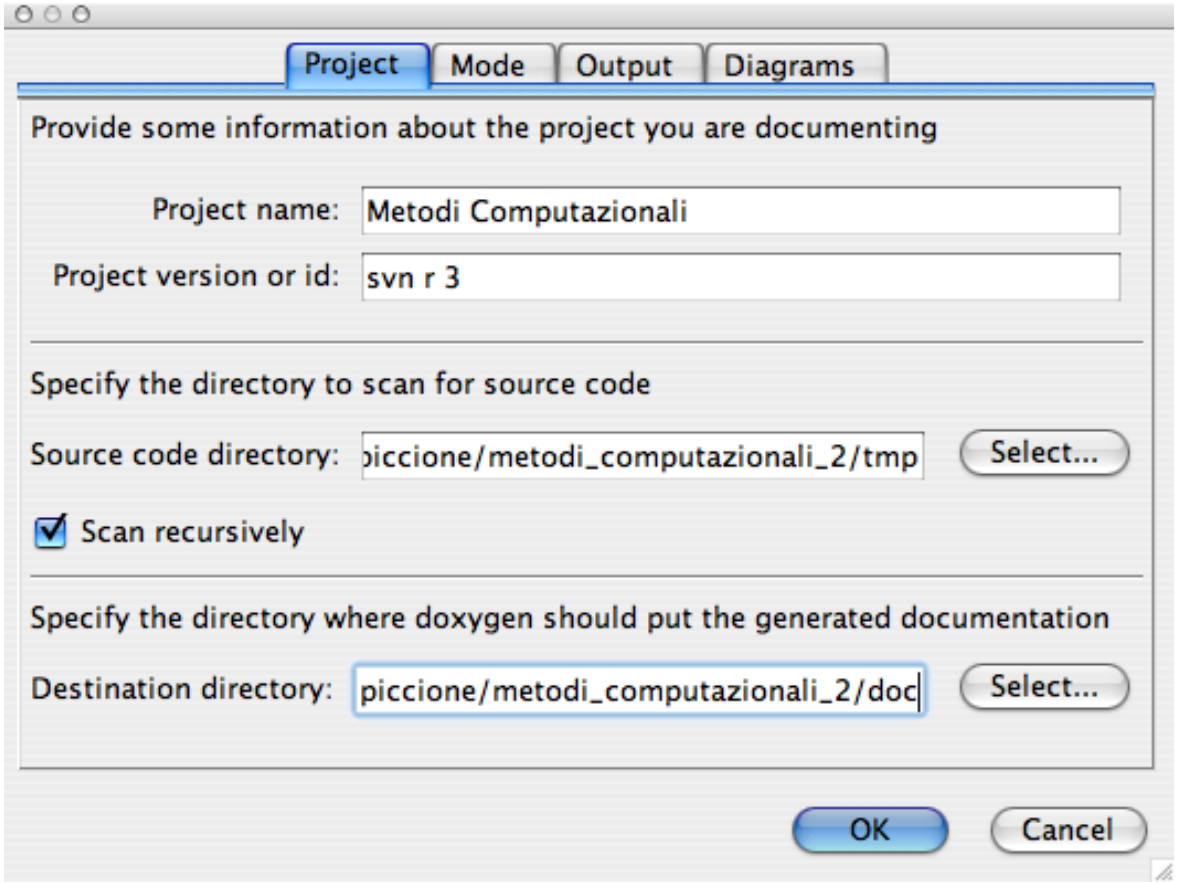
   e un errore casuale del 5%;
3. salvate i risultati con un formato [`id, x, y,` σ] e visualizzateli con gnuplot*.

```
*
pl "../data/sin.dat" u 2:3:4 t "Artificial data with noise" w e
repl 0.5+0.4*sin(6.28*x) t "Real function"
```

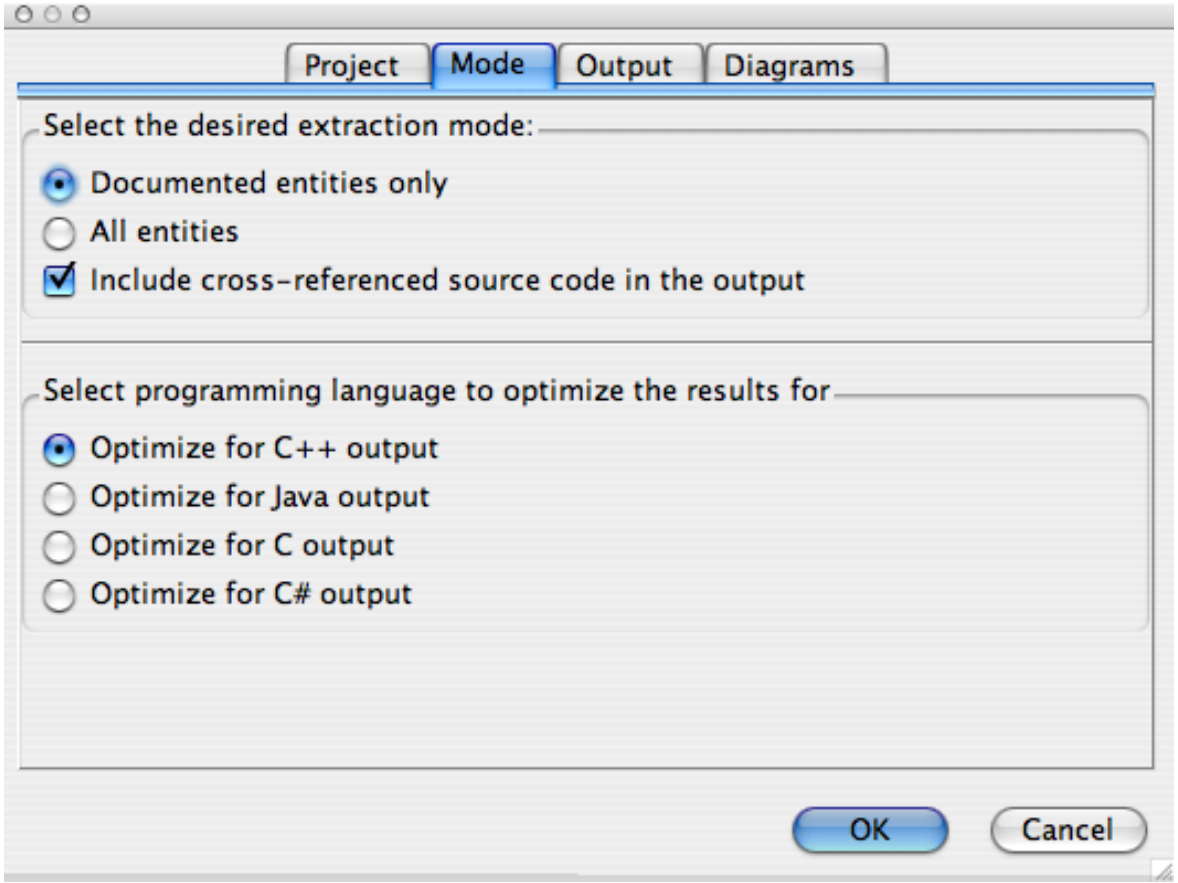
Doxygen GUI


Doxygen GUI frontend
Step 1: Configure doxygen
Choose one of the following ways to configure doxygen
Wizard...
Expert...
Load...
Step 2: Save the configuration file
Save...
Status: saved
Step 3: Specify the directory from which to run doxygen
Working directory:
/home/piccione/metodi_computazionali_2/tmp/doc
Select...
Step 4: Run doxygen
Start
Status: not running
Save log...
Output produced by doxygen

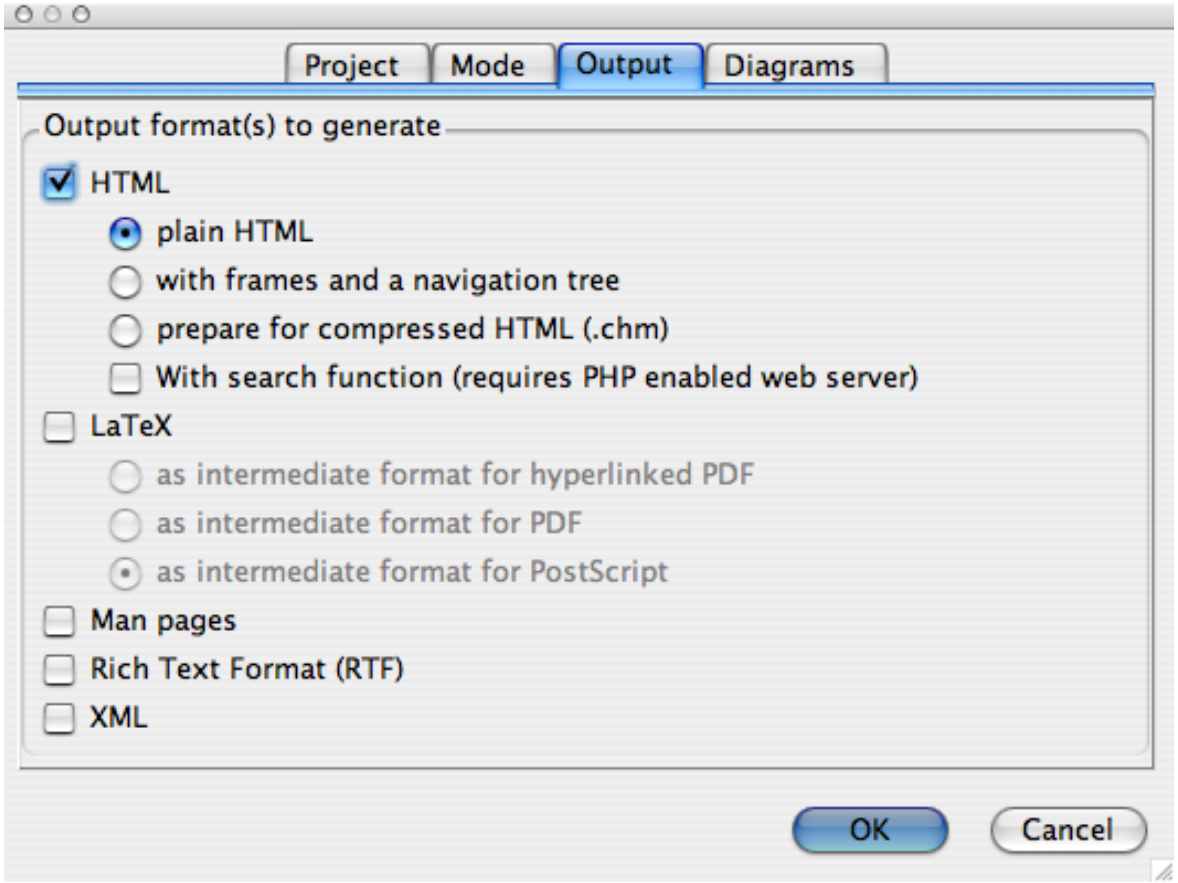
Doxygen Wizard



Doxygen Wizard



Doxygen Wizard



Riferimenti

- `http://www.doxygen.org`;
- `/home/piccione/metodi_computazionali_2/riferimenti/doxyfile`.

Esercizio

- create la documentazione per il codice scritto oggi;
- fate un commit del codice nella repository.