Τεχνολογικό Εκπαιδευτικό ίδρυμα Σερρών

Σχολή Τεχνολογικών Εφαρμογών

Τμήμα πληροφορικής και Επικοινωνιών

Πρότυπο συγγραφής κώδικα

Ζαχαρόπουλος Θεολόγος

Εισαγωγή

- ***Γιατί χρειάζεται ένα πρότυπο*** *:*

Τα περισσότερα λογισμικά αναπτύσσονται από ομάδες , έτσι τα πρότυπα υποχρεώνουν όλους τους συμμετέχοντες να οργανωθούν.
Βοηθούν την ομάδα να καταλάβει τι συμβαίνει στον κώδικά σας (τι γράψατε – γιατί το γράψατε).
Νέα μέλη καταλαβαίνουν γρήγορα τι γίνεται.

- ***Σε τι θα σας επηρεάσουν τα πρότυπα*** *:*

.Υποχρεωτική συμμόρφωση
.Στύλ σχολίων
.Ονοματολογία (Κλάσεων , μεταβλητών , συναρτήσεων)
.Δομές Ελέγχου
.Δομές Δεδομένων

- ***Μερικά γνωστά πρότυπα*** *:*

.Google : http://google-styleguide.googlecode.com/svn/trunk/cppguide.xml
.Gnu : http://www.gnu.org/prep/standards/standards.html
.Του Todd Hoff : http://www.possibility.com/Cpp/CppCodingStandard.html

- ***Το πρότυπο που θα ακολουθήσουμε*** *:*

Το πρότυπο που θα αναφερθεί είναι ένας συνδυασμός από το πρότυπου της google του Todd Heff επίσης μερικές συμβουλές από τις σημειώσεις του Κ. Πεταλίδη από το μάθημα Τεχνολογία Λογισμικού Ι (Τ.Ε.Ι. Σερρών).
Δεν θα μπω σε λεπτομέρειες θα αναφερθούν τα βασικότερα σημεία που είναι άξια της προσοχής μας.

Το Πρότυπο

- ***Γενικά*** *:*
  *.Σε κάθε δομή βάζουμε Braces ({ }) ακόμα και αν δε χρειάζεται.
  *.Όταν ελέγχετε για ισότητα το σταθερό μέρος πηγαίνει πάντοτε από αριστερά (π.χ. 0 == x) αυτό το κάνουμε γιατί περίπτωση απώλειας ενός '=' να μας ειδοποιήσει ο compiler.
  *.Προϋπολογίστε τιμές που μπορούν να προϋπολογιστούν.
  *.Φροντίστε τα σχόλια να συμφωνούν με τον κώδικα σας.
  *.Χρησιμοποιήστε δομές δεδομένων.
  *.Χρησιμοποιήστε βιβλιοθήκες.
  *.Γράψτε απλά και σωστά!

- ***Συναρτήσεις*** *:*

Τά ονόματα των συναρτήσεων πρέπει να δείχνουν την ενέργεια που κάνουν.

```
float getTemperature()
```

Αντι για

```
float temperature()
```

Τα ονόματα ξεκινούν πάντα με μικρό και αν περιέχουν περισσότερες από μία λέξεις κάθε λέξη ξεκινάει με κεφαλαίο γράμμα όπως παραπάνω.

Όταν κάποια συνάρτηση είναι τύπου bool τότε το όνομα της ξεκινάει από “is” με αυτό τον τρόπο αντιλαμβάνεται ο προγραμματιστής ότι είναι σαν σύνταξη πρότασης η εντολή.

```
If ( isCold() ) //Επιστρέφει TRUE όταν είναι κρύο
```

Πριν επιστραφεί κάποια τιμή από τις συναρτήσεις σας πρέπει να γίνεται πάντα έλεγχος της τιμής αυτής.
(π.χ. Αν κάποια συνάρτηση κάνει διαίρεση να ελέγχεται να μην γίνει ποτέ διαίρεση με το 0).

Ελέγξτε πάντα τις παραμέτρους που δέχονται οι συναρτήσεις σας.

Προγραμματίστε αμυντικά.

- ***Μεταβλητές*** *:*

Τα ονόματα των μεταβλητών πρέπει να δίνουν πλήρη πληροφορία σε τι αναφέρεται και ξεκινούν πάντα με μικρό και αν περιέχουν περισσότερες από μία λέξεις κάθε λέξη ξεκινάει με κεφαλαίο γράμμα και τα υπόλοιπα πεζά.

Οι global μεταβλητές ξεκινούν πάντα με "g".

```
float gTemp;
```

Οι global σταθερές έχουν όλα τα γράμματα κεφαλαία και οι λέξεις διαχωρίζονται με "_".

```
const A_GLOBAL_CONSTANT = 2;
```

Προτιμήστε να δηλώνετε σταθερές με την λέξη const παρά macros με την #define.

Οι μεταβλητές δείκτη ξεκινούν με "p".

```
char *pName;
```

Οι static μεταβλητές ξεκινούν με "s".

```
static int sCount;
```

Οι ENUM μεταβλητές έχουν όλα τα γράμματα κεφαλαία και οι λέξεις διαχωρίζονται με "_".

```
enum PinStateType
{
	PIN_OFF;
	PIN_ON;
};
```

- ***Σχόλια*** *:*

Υπάρχουν εργαλεία τα οποία κατά το δυνατόν αυτοματοποιούν την παραγωγή τεκμηρίωσης για ένα λογισμικό (π.χ. Doxygen , CppDoc).Αυτά δέχονται σαν είσοδο τον κώδικα ενός λογισμικού και αυτό παράγει τεκμηρίωση σε HTML.

Για να δουλέψει όμως σωστά πρέπει τα σχόλια στο πρόγραμμα να γράφονται με συγκεκριμένο τρόπο. Έτσι πρέπει να ακολουθούμε ένα πρότυπο σχολίων.

*.Όλα τα σχόλια μπαίνουν αμέσως πριν απο τον ορισμό της κλάσης , του constructor , destructor , των συναρτήσεων κτλ.

```
/**
   * This is the class comment for the class One
   */
class One
{
};
```

*.Εάν θέλετε σχόλια σε ορισμούς αυτού του τύπου :

```
  int x,y; //Comment for x and y
```

Καλύτερα γράψτε τα ως :

```
  int x; //Comments for x
   int y; //Comments for y
```

*.Μία προδιαγραφή :

1)Η πρώτη πρόταση στην περιγραφή μιας κλάσης , συνάρτησης κτλ. Πρέπει να είναι μια περίληψη ακριβής και συνοπτική.

2)Μετά την περίληψη έρχεται η κύρια περιγραφή. Περιγράφει συνοπτικά την λειτουργία της κλάσης , συνάρτησης κτλ.

3)Μετά την κύρια περιγραφή ακολουθούν οι ετικέτες. Περιγράφουν σημαντικά στοιχεία της της κλάσης , συνάρτησης κτλ.

@param : Για κάθε μία παράμετρο ορίζεται σχόλιο για τον ρόλο της.

@return : Ορίζει μία περιγραφή της τιμής που επιστρέφεται.

@throw/@exception : Περιγράφει αν η συνάρτηση μπορεί να καταλήξει σε μια εξαίρεση (Exception) και πότε συμβαίνει αυτό.

*.Παράδειγμα :

```
  /**
      * a normal member taking two arguments and returning an integer value.
      * @param a an integer argument.
      * @param s a constant character pointer.
      * @see Test()
      * @see ~Test()
      * @see testMeToo()
      * @return The test results
      */
      int testMe(int a,const char *s);
```

- ***Κλάσεις*** *:*

Το όνομα της κλάσης ξεκινάει πάντα με Κεφαλαίο και ακολουθούν πεζά.

```
class One
```

Όλα τα γράμματα είναι αγγλικά. Σε περίπτωση κλάσης με όνομα που περιέχει παραπάνω από μία λέξεις κάθε λέξη ξεκινάει με κεφαλαίο.

```
class OneTwoThree
```

Το όνομα της κλάσης αντιπροσωπεύει την οντότητα που θέλουμε να δηλώσουμε με μία λέξη.
Τα Braces ανοίγουν κάτω από το όνομα της κλάσης , και κλείνουν κάτω από την δήλωση της τελευταίας μεθόδου.

```
class One
{

};
```

Κάθε κλάση έχει ΠΑΝΤΑ : Constructors – Destructors – get – set

```
class One
{
private:
        int var;
public:
        One(void);
        One(var1);
        int getVar();
        void setVar();
        ~One(void);
};
```

- ***Header Files*** *:*

Κάθε header file .h στο οποίο θα γράφεται ο κώδικας δηλώσεων θα έχει και το αντίστοιχο .cpp αρχείο στο οποίο θα γράφεται ο κώδικας ανάπτυξης των κλάσεων – μεθόδων.

- Το .h αρχείο :
  Θα έχει πάντα το όνομα της κλάσεις με πεζά αγγλικά γράμματα , ξεκινάει πάντα με την δήλωση

```
#ifndef xx_h
#define xx_h
```

Ακολουθούν οι οδηγίες #include xxxx

Στη συνέχεια η δήλωση των κλάσεων – μεθόδων.
Και κλείνει με την οδηγία #endif

Παράδειγμα : (test.h)

```
/*
 * Contains Class Test
 */

#ifndef test_h
#define test_h

#include <iostream.h>

/*
 * This is a test class and it doesn't do something
 */
class Test
{
        private:
                int var;
        public:
                /**
                *Default Constructor
                 */
                Test(void);

                /** Copy Constructor
                 *
                 * @param var1 It is a test variable
                 */
                Test(var1);

                int getVar();
                void setVar(int var1);

                /**
                 * Destructor
                 */
                ~Test(void);
};

#endif
```

- Το .cpp αρχείο :
  Αρχικά εισάγουμε το αρχείο header στο αρχείο ανάπτυξης μας:
  #include "xx.h"

  Στην συνέχεια ακολουθεί η ανάπτυξη των συναρτήσεων.

  Παράδειγμα : (test.cpp)

```
#include "test.h"

Test::Test(void)
{
	var = 0;
}

Test::Test(int var1)
{
	var = var1;
}

int Test::getVar(void)
{
	return var;
}

void Test::setVar(int var1)
{
	var = var1;
}
```

– ***Επίλογος*** :

Παρακαλώ προσπαθήστε να κρατήσετε τον κώδικα σας σύμφωνο με το παραπάνω πρότυπο για να μην υπάρχει κανένα πρόβλημα και να είναι κώδικας σας πλήρες συνεργάσιμος.