ПМО ИУС

Оформление программного кода. Doxygen

Лекция 5

следящиедвиженияглазпрактическиполностьюавтомати
ческиеикакправиловозникаюттогдакогдастимулнаходит
сявдвиженииивотличиеотсаккадэтоплавныеисравнитель
номедленныедвиженияобычноихцельслежениезаобъек
томперемещающимсянанеподвижномфонеследователь
новданномслучаестимуломявляетсяскореенеместопол
ожениеобъектааскоростьегодвиженияискоростьследящ
ихдвиженийсоответствуетейэтимобеспечиваетсяотноси
тельностабильныйобразстимуланасетчаткедвижениегл
азсинхронизированноеперемещениемобъектаможетта
кжеспособствоватьболеечеткомувосприятиюформыпос
леднегоэтопроисходитпотомучтозрительнойсистемелег
чевоспринятьформустимулаеслиегопроекциянепереме
щаетсяпосетчаткеазафиксировананананей

следящиедвиженияглазпрактическиполностьюавтоматическиеикакправиловоз никаюттогдакогдастимулнаходитсявдвижениивотличиеотсаккадэтоплавныеиср авнительномедленныедвиженияобычноихцельслежениезаобъектомперемеща ющимсянанеподвижномфонеследовательновданномслучаестимуломявляется скореенеместоположениеобъектааскоростьегодвиженияискоростьследящихдв ижениийсоответствуетейэтимобеспечиваетсяотносительностабильныйобразсти муланасетчаткедвижениеглазсинхронизированноесперемещениемобъектамож еттакжеспособствоватьболеечеткомувосприятиюформыпоследнегоэтопроисхо дитпотомучтозрительнойсистемелегчевоспринятьформустимулаеслиегопроек циянеперемещаетсяпосетчаткеазафиксировананананей

следящие движения глаз практически полностью автоматические и как правило возникают тогда когда стимул находится в движении в отличие от саккад это плавные и сравнительно медленные движения обычно их цель слежение за объектом перемещащимся на неподвижном фоне следовательно в данном случае стимулом является скорее не местоположение объекта а скорость его движения и скорость следящих движений соответствует ей этим обеспечивается относительно стабильный образ стимула на сетчатке движение глаз синхронизированное с перемещением объекта может также способствовать более четкому восприятию формы последнего это происходит потому что зрительной системе легче воспринять форму стимула если его проекция не перемещается по сетчатке а зафиксирована на ней

следящие движения глаз практически полностью автоматические и как правило возникают тогда когда стимул находится в движении в отличие от саккад это плавные и сравнительно медленные движения обычно их цель слежение за объектом перемещающимся на неподвижном фоне следовательно в данном случае стимулом является скорее не местоположение объекта а скорость его движения и скорость следящих движений соответствует ей этим обеспечивается относительно стабильный образ стимула на сетчатке движение глаз синхронизированное с перемещением объекта может также способствовать более четкому восприятию формы последнего это происходит потому что зрительной системе легче воспринять форму стимула если его проекция не перемещается по сетчатке а зафиксирована на ней

Следящие движения глаз практически полностью автоматические и, как правило, возникают тогда, когда стимул находится в движении. В отличие от саккад это плавные и сравнительно медленные движения. Обычно их цель — слежение за объектом, перемещающимся на неподвижном фоне. Следовательно, в данном случае стимулом является скорее не местоположение объекта, а скорость его движения, и скорость следящих движений соответствует ей. Этим обеспечивается относительно стабильный образ стимула на сетчатке. Движение глаз, синхронизированное с перемещением объекта, может также способствовать более четкому восприятию формы последнего. Это происходит потому, что зрительной системе легче воспринять форму стимула, если его проекция не перемещается по сетчатке, а зафиксирована на ней.

ПОЛЯРНАЯ ПРАВДА

Присяга родине

УКАЗ

ПРЕЗИДИУМА ВЕРХОВНОГО СОВЕТА СССР

О тексте военной присяги Рабоче-Крестьянской Красной Армии

УКАЗ

ПРЕЗИДИУМА ВЕРХОВНОГО СОВЕТА СССР

О тексте военной присяги Рабоче-Крестьянского Военно-Морского Флота

УКАЗ

ПРЕЗИДИУМА ВЕРХОВНОГО СОВЕТА СССР

О порядке принятия военной присяги

Цели оформления программного кода

1. Повышение качества сопровождения.
2. Облегчение восприятия и чтения кода.
3. При работе в команде общение идет на одном языке.

Комментарии

```
// Вычисление суммы ряда:
// (x+1)/x^2  + (x+2)/x^2 + ..
// + (x+n)/x^2.
float
row_calc(float x, int n)
{
    // Если x = 0,
    if ( x == 0 )
    {
        // то при вычислении будет деление на 0.
        // Сигнализируем об ошибке.
        throw std::exception("Invalid arg X.");
    }

    // Хранение результата.
    // Сумма сначала равна 0.
    float result = 0;

    // Обработать n слогаемых.
    for(int i = 1; i < n; i++)
    {
        result += (x+1) / (x*x);
    }

    return result;
}
```

```
int
main()
{
    // Ввод данных.

    // Аргумент расчета.
    float x;
    // Количество элементов ряда.
    int n;

    std::cout << "Enter X, N:";
    std::cin >> x >> n;

    // Расчет.

    float result = row_calc(x, n);

    // Печать результата.

    std::cout <<
        "Result:" << result << std::endl;

    return 0;
}
```

Комментарии

```
int x;          //< Координата по горизонтали.
int y;          //< Координата по вертикали.
int number;     //< Количество элементов.
```

Комментарии

```
func(int x)
/*
   Не располагайте комментарий здесь.
*/
{
   //..
}

/*
   Лучше здесь.
*/
func(int x)
{
   /*
        Или здесь.
   */
}
```

Комментарии

```
int i = 0;
while(i < n)
{
   ++i;
   for(int j = 0; j < m; ++j)
   {
       if (a[i][j] > 0)
       {
               …
       }

   } // for

   i += a[i][j];

} // while(i < n)
```

Форматирование

```
0    1    2    3    4
int main()
{
…

    int i = 0;
    while(i < n)
    {
        ++i;
        for(int j = 0; j <  m; ++j)
        {

            if (a[i][j] > 0)
            {
                sum += a[i][j];
            }
        } // for

        i += a[i][j];

    } // while(i < n)
…
}
```

Многострочное форматирование

```
if (
   (x > 0) && (y > 0) &&
   (x < n) && (y < n) )
{

}

// Печать прямого порядка.
std::cout <<
   "All words in enter order:“ <<
   all_output.str() <<
   std::endl;
```

Форматирование { }

```
if (x > 0)
{
  …
}
```

Пробелы и скобки

```
// Вычисление суммы ряда:
// (x+1)/x^2  + (x+2)/x^2 + ..
// + (x+n)/x^2.
float
row_calc( float x, int n )
{
    // Если x = 0,
    if ( x == 0 )
    {
        // то при вычислении будет деление на 0.
        // Сигнализируем об ошибке.
        throw std::exception( "Invalid arg X.“ );
    }

    // Хранение результата.
    // Сумма сначала равна 0.
    float result = 0;

    // Обработать n слогаемых.
    for(int i = 1; i < n; i++)
    {
        result += (x + 1) / (x * x);
    }

    return result;
}
```

Венгерская нотация

Учет факторов при введении идентификаторов:

1. Мнемоническое значение.
2. Смысловое значение.
3. Преемственность.
4. Скорость решения.

Венгерская нотация

Подход дает:

1. Мнемонические названия в строгом смысле.
2. Названия несут в себе смысловое значение идентификатора.
3. Названия не противоречивы.
4. Построение названий делается механически и как результат быстро.

Венгерская нотация - правила

1. Описание характеристики идентификатора входит в идентификатор.
2. Названия отличают идентификаторы, имеющие один и тот же тип и существующие в одном контексте.
3. Простые типы названы короткими тегами.
4. Составные части пишутся с большой буквы, всё выражение пишется слитно (Camel Case).

Примеры (описание характеристики, название):

rowFirst

dwHeadersLength

lpOptional

hEnumHandle

GetURLConnection

Нотация Lower Case

Венгерская	**Lower Case**
rowFirst	row_first
dwHeadersLength	headers_length
lpOptional	optional
hEnumHandle	enum_handle
GetURLConnection	get_url_connection

Lower Case

```
// Несколько самостоятельных названий.
int first_table__database_check;

// Использование постфиксов.
point_t point;

// Использование префиксов.
const int c_max_store = 100;
```

Файлы

header_file.hpp

source_file.cpp

my_class.hpp

my_class.cpp

```
// Работа со строками.
#include <string>

// Динамические объекты.
#include <vector>
#include <list>

// Библиотеки ввода-вывода.
#include <iostream>
#include <iomanip>
```

```
#include <library/h/my_lib.hpp>
#include <laba3/h/main.hpp>
#include <project/db_connect/h/odbc.hpp>
```

Типы

```
struct  my_struct_t {...};
typedef unsigned char uint8_t;
class   io_subsystem_t { ... };
class math_ex_t { … };
```

Функции

```
// Без аргументов описывается без void.
void
some_fuction();

void
init_io_subsystem__iostream( ostream & o );
```

Автоматическая система документирования Doxygen

```
/*! Вычисление суммы ряда: (x+1)/x^2  + (x+2)/x^2 + ..  + (x+n)/x^2.

    \param x аргумент расчета.
    \param n число итераций.
    \return сумма ряда.
    \throw std::exception в случае, если ошибка расчета.

    \note аргумент x должен быть отличен от 0.
*/
float
row_calc( float x, int n )
{
    // Если x = 0,
    if ( x == 0 )
    {
        // то при вычислении будет деление на 0. Сигнализируем об ошибке.
        throw std::runtime_error( "Invalid arg X.“ );
    }

    // Хранение результата. Сумма сначала равна 0.
    float res = 0;

    // Обработать n слогаемых.
    for( int i = 1; i < n; ++i )
    {
        res += (x+1) / (x*x);
    }

    return res;
}
```

Комментарии Doxygen

Два вида комментариев:
1. Краткий (brief).
2. Полный (detailed).

```
/**
   Полный комментарий.
*/

/*!    Полный комментарий.*/

///    Краткий комментарий.

//!    Краткий комментарий.

/*!
   \class Test
   \brief Тестовый класс.

   Полное описание класса.
*/
```

Порядок комментариев

```
/*! Вычисление суммы ряда:
 (x+1)/x^2  + (x+2)/x^2 + ..  + (x+n)/x^2.

   \return сумма ряда.
   \throw std::exception в случае, если ошибка расчета.

   \note аргумент x должен быть отличен от 0.
*/
float
row_calc(
   //! аргумент расчета.
   float x,
   //! число итераций.
   int n ) {
…
```

```
int var; /*!< Подробный комментарий. */
```

Описание идентификаторов

\struct	**Документирует структуру.**
\union	**Документирует объединение.**
\enum	**Документирует перечисление.**
\fn	**Документирует функцию.**
\var	**Документирует переменную.**
\def	**Документирует макрос подстановки *#define*.**
\file	**Документирует файл.**
\namespace	**Документирует пространство имен.**

Команды оформления

```
\param описание параметров функции.
\return описание возвращаемого значения.
\retval описание конкретного возвращаемого значения


\attention \warning \note
\since \author \version \date
\exception \throw
\mainpage \page \subpage \section \subsection \paragraph
```