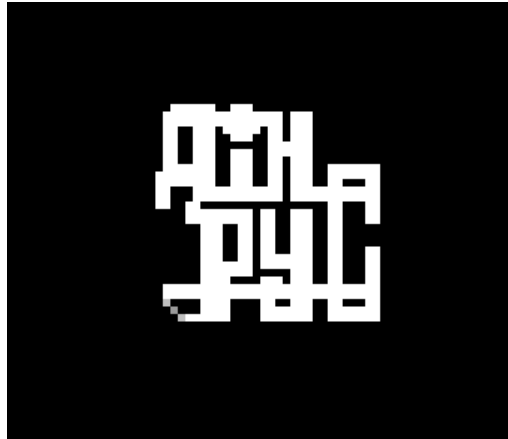


## КРАТКОЕ, ПРАКТИЧНОЕ ВВЕДЕНИЕ В ДИНРУС



\*\*\*\*\*

```
/**/ Главный рантаймный модуль языка программирования Динрус,  
* поддерживающий совместимость с английской версией.  
* Разработчик Виталий Кулич  
*/  
module object;  
public import base;
```

\*\*\*\*\*

Пожалуй, вся работа Динрус начинается с этого модуля.

Следует заметить, что он публично импортирует другой модуль, с названием `base`. Именно в базовом модуле помещены все основные типы, применяемые в программировании на этом языке!

Это означает, что в начале любой работы компилятором автоматически загружаются все основные типы, дающие далее возможность менее вдумчивой работы относительно указания ссылок на пакеты, в которых следует искать нужные типы. (Такая неприятность хорошо известна по языку C++!)

Как видим, далее идёт класс Объект. В нём оставлена двуязычность. Например, `void dispose()` и проц вывести() выполняют одну и ту же работу. Однако, в компиляторе DMD использована такая интересная хитрость, о которой следует знать как можно скорее! -

DMD ЖДЁТ некоторых заранее условленных данных, разрабатывать которые можно в дальнейшем, при условии, что имена этих классов или методов будут соответствовать ожидаемому компилятором.

Так, например, для всех классов этого модуля оставлены АНГЛИЙСКИЕ ИМЕНА. Хотя с помощью важной хитрости класс `Object` одновременно становится и классом Объект.

Хитрость эта скрывается...

```
extern (D) class Object  
{  
  
    void dispose();  
    проц вывести();  
  
    void print();  
    проц вывести();  
  
    string toString();
```

```

ткст вТкст();

hash_t toHash();
т_хэш вХэш();

int opCmp(Object o);
int opEquals(Object o) ;

    interface Monitor
    {
    void lock();
        alias lock блокируй;
    void unlock();
        alias unlock разблокируй;
    }
    alias Monitor Монитор;

final проц notifyRegister(void delegate(Object) дг);
final проц уведомиРег(проц delegate(Объект) дг);

final проц notifyUnRegister(void delegate(Object) дг);
final проц уведомиОтрег(проц delegate(Объект) дг);

static Object factory(string classname);
static Объект фабрика(ткст имякласса);

}

ЗДЕСЬ!!!->>>

alias Object Объект;
alias Object.Monitor IMonitor, ИМонитор;

```

Ровно таким же образом, с помощью волшебного слова **alias**, **int** в Динрусе превращается в **цел**, **string** в **ткст**, **ubyte** в **ббайт**, **false** в **нет**, **true** в **да**, **null** в **пусто** и т.д. Это хорошая находка, чего не хватает, например в том же С#...

Рассмотрим другой момент:

```

/**
 * All recoverable exceptions should be derived from class Exception.
 */
class Exception : Object

/**
 * All irrecoverable exceptions should be derived from class Error.
 */
class Error : Exception

```

В справочной системе Phobos указано, что все восстанавливаемые исключения должны происходить от класса Исключение, а все невозстанавливаемые - наследовать от класса Ошибка.

Сам класс Ошибка происходит от класса Исключение, о чём говорит такой синтаксис: **class Error : Exception**. Двоеточие - это признак того, что первый класс наследует от второго. В Динрусе всё как и в D v1: множественное наследование недопустимо, как это возможно в других языках.

С моей т.з. наличие класса Ошибка совершенно необоснованно. Достаточно и одного класса для этой информационной цели. Ошибка - это когда исключение происходит в форме сбоя, т.е. об этом уже невозможно никак сообщить. Поэтому и излишне создавать такой класс.

И наследовать от исключения ошибка никак уж не может!... Если в программирование и не должно быть никакой философии, то хотя бы логика!... Вкратце, замечу: КРИТИКА и ещё раз КРИТИКА. Программисты, как и сапёры, ошибаются только раз. После чего у них же самих возникает масса недоразумений и ляпов.

Исключения следует собрать в одном модуле, exception. Такая строгость просто необходима, так как эйфория написания лишнего кода кажется глупой и нецелесообразной. Словом, сам себе на уме подразумеваю, что класс Ошибка надо бы пометить как deprecated. В Руладе он есть, так как нужен для некоторых модулей, написанных не мною. В Динрусе его следует самого исключить)))

После двоеточия может быть указан, один класс, один интерфейс. Правда,

иногда, интерфейсов несколько... это, кажется, допустимо, реализовывать несколько интерфейсов в одном и том же классе... Недопустимо только наследование от нескольких классов...

Итак, поскольку информация с английскими символами нужна чисто компилятору, для русского программиста в итоге в коде класса Объект должно видаться следующее:

```
класс Объект
{
    проц вывести(); //Выкидывает из карты памяти?
    проц вывести(); //Выдаёт то же, что и вТкст()?
    ткст вТкст(); //Возвращает имя класса?
    т_хэш вХэш(); //Выводит хэш-код Объекта?

    оператор цел opCmp(Объект о); // Это сравнение, т.е. ==?
    оператор цел opEquals(Объект о); //А это сравнение, т.е. тоже ==?!
    //Видимо, без дополнительного анализа исходного кода
    //едва ли удастся разобраться...

    интерфейс Монитор
    {
        проц блокируй(); //Блокирует объект?
        проц разблокируй(); //Разблокирует объект?
    }

    финальная проц уведомиРег(проц делегат(Объект) дг); //Регистрирует объект?
    финальная проц уведомиОтрег(проц делегат(Объект) дг); //Снимает регистрацию?
    статическая Объект фабрика(ткст имякласса); //Порождает такой же объект с другим именем?
    //При очень критическом анализе возникает ещё масса вопросов типа зачем это надо и как оно
    //работает? Се ля ви... В этом-то и вся соль!(((
}
```

Все функции типа **opXXXXXXXX()** являются специальными, обусловленными компилятором, - вернее его программой, - операторами, о чём говорилось ранее.

Их список достаточно обширен, поэтому найти его можно будет в специальных таблицах, составление которых представляется весьма необходимым шагом в дальнейшем формировании справочной базы для языка программирования Динрус. Эти функции являются определениями операторов, например, сравнения, присвоения, деления, умножения и т.д. одного Объекта по отношению к другому Объекту.

Далее оставляю листинг, как он есть. Будем считать, что это было введением в саму суть программирования на Динрусе, т.е. сильно модифицированном языке D. Да, он является Динрусом, при этом оставаясь D v1!

#### Листинг модуля object \*\*\*\*\*

```
/**
 * Главный рантаймный модуль языка программирования Динрус,
 * поддерживающий совместимость с английской версией.
 * Разработчик Виталий Кулич
 */
module object;
public import base;

extern (D) class Object

{
    проц dispose();
    проц вывести();
    проц print();
    проц вывести();

    ткст toString();
    ткст вТкст();

    hash_t toHash();
    т_хэш вХэш();

    int opCmp(Object o);
    int opEquals(Object o);
    интерфейс Monitor
    {
        проц lock(); alias lock блокируй;
        проц unlock(); alias unlock разблокируй;
    }
    alias Monitor Монитор;

    final проц notifyRegister(проц delegate(Object) дг);
    final проц уведомиРег(проц delegate(Объект) дг);

    final проц notifyUnRegister(проц delegate(Object) дг);
    final проц уведомиОтрег(проц delegate(Объект) дг);
}
```

```

static Object factory(ткст classname);
    static Объект фабрика(ткст имякласса);
}
alias Object Объект;
alias Object.Monitor IMonitor, ИМонитор;

ИнфоКлассе дайИоК(Объект о){return о.classinfo ;}

//ИнфоКлассе дайИоК(Объект о){return о.classinfo;}

////////////////////////////////////
/**
 * Все восстановимые исключения должны происходить от класса Исключение.
 */
extern (D) class Exception : Object
{
    ткст msg; alias msg сооб;
    ткст file; alias file файл;
    size_t line;    alias line строка;
    TraceInfo info;    alias info инфо;
    Exception next;    alias next следщ;
    struct FrameInfo
    {
        long line;    alias line строка;
    }
    size_t iframe;    alias iframe икадр;
    ptrdiff_t offsetSymb;    alias offsetSymb симвСмещ;
    size_t baseSymb;    alias baseSymb симВовы;
    ptrdiff_t offsetImg;    alias offsetImg обрСмещ;
    size_t baseImg;    alias baseImg обрОвы;
    size_t address;    alias address адрес;
    ткст file;    alias file файл;
    ткст func;    alias func функц;
    ткст extra;    alias extra экстра;
    bool exactAddress;    alias exactAddress точныйАдрес;
    bool internalFunction;    alias internalFunction внутрФункция;
    alias проц function(FrameInfo*, проц delegate(char[])) FramePrintHandler,
    ОбработчикПечатиКадра;
    static FramePrintHandler defaultFramePrintingFunction;
    alias defaultFramePrintingFunction дефФцияПечатиКадра;
    проц writeOut(проц delegate(char[]) sink);
    проц выпиши(проц delegate(ткст) синк);
    проц clear();
    проц сотри();
}
alias FrameInfo ИнфоОКадре;//
interface TraceInfo
{
    int opApply( int delegate( ref FrameInfo fInfo ) );
    проц writeOut(проц delegate(char[]) sink);
    alias writeOut выпиши;
}
alias TraceInfo ИнфоОСледе;//
this( ткст сооб, ткст file, long line, Exception next, TraceInfo info );
this( ткст сооб, Exception next=null );
this( ткст сооб, ткст file, long line, Exception next=null );
override проц print();
override проц выведи();
override ткст toString();
override ткст вТкст();
/+
проц writeOutMsg(проц delegate(char[]) sink);
проц выпишиСооб(проц delegate(ткст) синк);
проц writeOut(проц delegate(char[]) sink);
проц выпиши(проц delegate(ткст) синк);
+/-
проц сбрось();
}
alias Exception Исключение, Искл;
////////////////////////////////////

alias Исключение.ИнфоОСледе function( ук укз = пусто ) TraceHandler, Следопыт;

private Следопыт следопыт = пусто;
////////////////////////////////////

/**
 * Все нововосстановимые исключения должны происходить от класса Ошибка.
 */
extern (D) class Error : Exception
{
    Error next; alias next следщ;
    ткст msg; alias msg сооб;
    override проц print();
    override проц выведи();
    override ткст toString();
    override ткст вТкст();
}

```

```

/**
 * Конструктор; сооб - сообщение, описывающее исключение.
 */
    this(текст сооб);
this(текст сооб, Error next);
}

alias Error Ошибка, Ош;
////////////////////////////////////

alias проц delegate(Object) DEvent, ДСобыт;

extern (D) struct Monitor
{
    проц delegate(Object)[] delegates;
    extern(C) extern IMonitor impl;
    extern(C) extern ДСобыт[] devt;
}
alias Monitor Монитор;

/*****
 * Информация о каждом модуле.
 */
alias ModuleInfo ИнфоМодуле;
extern(D) class ModuleInfo
{
    extern(C) extern char name[];
    extern(C) extern ИнфоМодуле importedModules[];
    extern(C) extern ИнфоКлассе localClasses[];

    extern(C) extern бцел flags; // initialization state

    проц function() ctor; // module static constructor (order dependent)
    проц function() dtor; // module static destructor
    проц function() unitTest;
    /*проц (*ctor)(); // module static constructor (order dependent)
    проц (*dtor)(); // module static destructor
    проц (*unitTest)(); // module unit tests*/

    extern(C) extern ук xgetMembers; // module getMembers() function

    проц function() ictor; //проц (*ictor)(); // module static constructor (order
independent)

    static int opApply( int delegate( ref ModuleInfo ) др );
}
/*****
 * Возвращает коллекцию всех модулей в программе.
 */
static ИнфоМодуле[] модули();
}

extern(D) class ОшКтораМодуля : Исключение
{
    this(ИнфоМодуле m);
}

////////////////////////////////////

extern (C) struct Interface
{
    extern(C) extern ИнфоКлассе classinfo; alias classinfo классинфо;
    extern(C) extern ук [] vtbl; alias vtbl вирттаб;
    extern(C) extern цел offset; alias offset смещение;
}
alias Interface Интерфейс;
////////////////////////////////////

alias ClassInfo ИнфоКлассе;
extern (D) class ClassInfo
{
    extern(C) extern byte[] init; alias init иниц;
    byte[] getSetInit(byte[] init = null);
    байт[] дайУстИниц(байт[] иниц = пусто);
    extern(C) extern текст name; alias name имя;
    текст getSetName(текст name = null);
    текст дайУстИмя(текст имя = пусто);

    extern(C) extern ук [] vtbl; alias vtbl вирттаб;
    ук[] getSetVtbl(ук[] vtbl = null);
    ук[] дайУстВирттаб(ук[] вирттаб = пусто);

    extern(C) extern Interface[] interfaces; alias interfaces интерфейсы;
    Interface[] getSetInterfaces(Interface[] interfaces = null);
    Интерфейс[] дайУстИнтерфейсы(Интерфейс[] интерфейсы = пусто);

    extern(C) extern ClassInfo base; alias base основа;
    ИнфоКлассе getSetBase(ИнфоКлассе base = null);
    ИнфоКлассе дайУстОву(ИнфоКлассе основа = пусто);

```

```

extern(C) extern ук destructor;    alias destructor деструктор;
    ук getSetDestructor(ук destructor = null);
    ук дайУстДестр(ук деструктор = пусто);

проц (*classInvariant)(Object);

extern(C) extern бцел flags;    alias flags флаги;
    // 1:                // ИИнокогнито (IUnknown)
    // 2:                // нет возможных указателей на память см
    // 4:                // есть члены offTi[]
    // 8:                // есть конструкторы
    // 32:               // есть инфотипе
    бцел getSetFlags(бцел flags = бцел.init);

extern(C) extern ук deallocator;    alias deallocator выместитель;
    ук getSetDeallocator(ук deallocator = null);
    ук дайУстДеаллок(ук выместитель = пусто);

extern(C) extern OffsetTypeInfo[] offTi;    alias offTi смТи;
    OffsetTypeInfo[] getSetOffTi(OffsetTypeInfo[] offTi = null);
    OffsetTypeInfo[] дайУстСмТи(ИнфОТипеИСмещ[] смТи = пусто);

проц function(Object) defaultConstructor;

extern(C) extern TypeInfo typeinfo; alias typeinfo инфотипе;
    ИнфОТипе getSetTypeInfo(ИнфОТипе typeinfo = null);
    ИнфОТипе дайУстИнфОТипе(ИнфОТипе инфотипе = пусто);

static ClassInfo find(ткст classname);
    static ИнфОКлассе найди (ткст имякласса);

    Object create();
    Объект создай();
}

////////////////////////////////////
extern (C) struct OffsetTypeInfo
{
    extern(C) extern size_t offset;    alias offset смещение;
    extern(C) extern TypeInfo ti;    alias ti иот;
}
alias OffsetTypeInfo ИнфОТипеИСмещ;

////////////////////////////////////

alias TypeInfo ИнфОТипе;
extern (D) class TypeInfo
{
    hash_t toHash();
    т_хэш вХэш();

    override int opCmp(Object o);
    override int opEquals(Object o);

    hash_t getHash(in ук p);
    т_хэш дайХэш(in ук p);
    int equals(in ук p1, in ук p2) ;
    цел равны(in ук p1, in ук p2);

    int compare(in ук p1, in ук p2) ;
    цел сравни(in ук p1, in ук p2);
    size_t tsize();
    т_мера тразм();
    проц swap(ук p1, ук p2);
    проц поменяй(ук p1, ук p2);

    TypeInfo next();
    ИнфОТипе следщ();
    проц[] init();
    проц[] иниц();
    бцел flags();
    бцел флаги();
    OffsetTypeInfo[] offTi();
    ИнфОТипеИСмещ[] смТи();
}
////////////////////////////////////
extern (D) class TypeInfo_Typedef : ИнфОТипе
{
    override ткст toString();
    override ткст вТкст();

    override int opEquals(Object o);

    override hash_t getHash(in ук p) ;
    override т_хэш дайХэш(in ук p);
    override int equals(in ук p1, in ук p2) ;
    override цел равны(in ук p1, in ук p2);
    override int compare(in ук p1, in ук p2) ;
    override цел сравни(in ук p1, in ук p2);

```

```

        override size_t tsize();
        override т_мера тразм();
        override проц swar(ук p1, ук p2) ;
        override проц поменяй( ук p1, ук p2);
        override ИнфОТипе next() ;
        override ИнфОТипе следщ();
        override бцел flags() ;
    override бцел флаги();
    override проц[] init() ;
        override проц[] иниц();

extern(C) extern ИнфОТипе base;    alias base основа;
ИнфОТипе getSetBase(ИнфОТипе base = null);
ИнфОТипе дайУстОву(ИнфОТипе основа = пусто);
extern(C) extern ткст name;    alias name имя;
ткст getSetName(ткст name = null);
ткст дайУстИмя(ткст имя = пусто);
extern(C) extern проц[] m_init;
}
alias TypeInfo_Typedef ТипТипдеф;
////////////////////////////////////

extern (D) class TypeInfo_Enum : TypeInfo_Typedef
{

}
alias TypeInfo_Enum ТипПеречень;
////////////////////////////////////

extern (D) class TypeInfo_Pointer : ИнфОТипе
{
    override ткст toString() ;
        override ткст вТкст();
        override int opEquals(Object o);

    hash_t getHash(ук p);
        т_хэш дайХэш(ук p);

    int equals(ук p1, ук p2);
        цел равны(ук p1, ук p2);
        int compare(ук p1, ук p2);
        цел сравни(ук p1, ук p2);

    override size_t tsize();
        override т_мера тразм();

    override проц swar(ук p1, ук p2);
        override проц поменяй( ук p1, ук p2);

    override ИнфОТипе next() ;
        override ИнфОТипе следщ();
        override бцел flags();
        override бцел флаги();

    extern(C) extern ИнфОТипе m_next;
}
alias TypeInfo_Pointer ТипУказатель;
////////////////////////////////////

extern (D) class TypeInfo_Array : ИнфОТипе
{
    override ткст toString() ;
        override ткст вТкст();

    override int opEquals(Object o);

    hash_t getHash(ук p);
        override т_хэш дайХэш(ук p);

    int equals(ук p1, ук p2);
        цел равны(ук p1, ук p2);

    int compare(ук p1, ук p2);
        цел сравни(ук p1, ук p2);
        override size_t tsize();
        override т_мера тразм();

    override проц swar(ук p1, ук p2);
        override проц поменяй( ук p1, ук p2);

    extern(C) extern ИнфОТипе value;

    override ИнфОТипе next() ;
        override ИнфОТипе следщ();

    override бцел flags();
        override бцел флаги();
}
alias TypeInfo_Array ТипМассив;

```

```

////////////////////////////////////
extern (D) class TypeInfo_StaticArray : ИнфоТипе
{
    override ткст toString();
    override ткст вТкст();

    override int опEquals(Object o);

    override hash_t getHash(in ук p);
    override т_хэш дайХэш(in ук p);
    override int equals(in ук p1, in ук p2);
    override цел равны(in ук p1, in ук p2);

    override int compare(in ук p1, in ук p2);
    override цел сравни(in ук p1, in ук p2);

    override size_t tsize();
    override т_мера тразм();

    override проц swap(ук p1, ук p2);
    override проц поменяй( ук p1, ук p2);

    override проц[] init() ;
    override проц[] иниц();
    override ИнфоТипе next() ;
    override ИнфоТипе следщ();
    override бцел flags();
    override бцел флаги();

    extern(C) extern ИнфоТипе value;    alias value значение;
    ИнфоТипе getSetValue(ИнфоТипе value = null);
    ИнфоТипе дайУстЗначение(ИнфоТипе значение = пусто);
    extern(C) extern size_t len;    alias len длин;
    т_мера getSetLength(т_мера len = т_мера.init);
    т_мера дайУстДлину(т_мера длин = т_мера.init);
}
alias TypeInfo_StaticArray ТипСтатМас;

////////////////////////////////////
//
extern (D) class TypeInfo_AssociativeArray : ИнфоТипе
{

    override ткст toString();
    override ткст вТкст();

    override /*int*/ int опEquals(Object o);

    override hash_t getHash(in ук p);
    override т_хэш дайХэш(in ук p);

    override size_t tsize();
    override т_мера тразм();
    override int equals(in ук p1, in ук p2);
    override цел равны(in ук p1, in ук p2);

    override int compare(in ук p1, in ук p2);
    override цел сравни(in ук p1, in ук p2);

    override ИнфоТипе next() ;
    override ИнфоТипе следщ();
    override бцел flags() ;
    override бцел флаги();

    extern(C) extern ИнфоТипе value;    alias value значение;
    ИнфоТипе getSetValue(ИнфоТипе value = null);
    ИнфоТипе дайУстЗначение(ИнфоТипе значение = пусто);
    extern(C) extern ИнфоТипе key;    alias key ключ;
    ИнфоТипе getSetKey(ИнфоТипе key = null);
    ИнфоТипе дайУстКлюч(ИнфоТипе ключ = пусто);
}
alias TypeInfo_AssociativeArray ТипАссоцМас;

////////////////////////////////////
//

extern (D) class TypeInfo_Function : ИнфоТипе
{
    override ткст toString();
    override ткст вТкст();

    override int опEquals(Object o);
    override size_t tsize();
    override т_мера тразм();

    extern(C) extern ИнфоТипе next;    alias next следщ;
    ИнфоТипе getSetNext(ИнфоТипе next = null);
    ИнфоТипе дайУстСледщ(ИнфоТипе следщ = null);
}

```



```

alias TypeInfo_Function ТипФункция;
////////////////////////////////////

extern (D) class TypeInfo_Delegate : ИнфоТипе
{
    override ткст toString();
    override ткст вТкст();

    override int опEquals(Object o);

    override size_t tsize();
    override т_мера тразм();

    override бцел flags();
    override бцел флаги();

    extern(C) extern ИнфоТипе next;    alias next следщ;
    ИнфоТипе getSetNext(ИнфоТипе next = null);
    ИнфоТипе дайУстСледщ(ИнфоТипе следщ = null);
}
alias TypeInfo_Delegate ТипДелегат;
////////////////////////////////////

extern (D) class TypeInfo_Class : ИнфоТипе
{
    override ткст toString();
    override ткст вТкст();

    override int опEquals(Object o);

    hash_t getHash(ук p);
    override т_хэш дайХэш(ук p);

    int equals(ук p1, ук p2);
    цел равны(ук p1, ук p2);

    int compare(ук p1, ук p2);
    цел сравни(ук p1, ук p2);

    override size_t tsize();
    override т_мера тразм();

    override бцел flags();
    override бцел флаги();

    override OffsetTypeInfo[] offTi();
    override ИнфоТипеИСмеш[] смТи();

    extern(C) extern ClassInfo info;    alias info инфо;
    ИнфоКлассе getSetInfo(ИнфоКлассе info = null);
    ИнфоКлассе дайУстИнфо(ИнфоКлассе инфо = пусто);
}
alias TypeInfo_Class ТипКласс;
////////////////////////////////////
/
extern (D) class TypeInfo_Interface : ИнфоТипе
{
    override ткст toString();
    override ткст вТкст();

    override int опEquals(Object o);

    hash_t getHash(ук p);
    т_хэш дайХэш(ук p);

    int equals(ук p1, ук p2);
    цел равны(ук p1, ук p2);

    int compare(ук p1, ук p2);
    цел сравни(ук p1, ук p2);

    override size_t tsize();
    override т_мера тразм();

    override бцел flags();
    override бцел флаги();

    extern(C) extern ClassInfo info;    alias info инфо;
    ИнфоКлассе getSetInfo(ИнфоКлассе info = null);
    ИнфоКлассе дайУстИнфо(ИнфоКлассе инфо = пусто);
}
alias TypeInfo_Interface ТипИнтерфейс;
////////////////////////////////////

extern (D) class TypeInfo_Struct : ИнфоТипе
{
    override ткст toString();
    override ткст вТкст();

    override int опEquals(Object o);

```

```

hash_t getHash(ук p);
    т_хэш дайХэш(ук p);

int equals(ук p1, ук p2);
    цел равны(ук p1, ук p2);

int compare(ук p1, ук p2);
    цел сравни(ук p1, ук p2);

override size_t tsize();
    override т_мера тразм();

override проц[] init();
    override проц[] иниц();

override бцел flags();
    override бцел флаги();

extern(C) extern ткст name;    alias name имя;
    ткст getSetName(ткст name = null);
    ткст дайУстИмя(ткст имя = пусто);
    extern(C) extern проц[] m_init;
    hash_t function(проц*) xtoHash;
int function(проц*,проц*) хорEquals;
int function(проц*,проц*) хорСмп;
ткст function(проц*) xtoString;

extern(C) extern бцел m_flags;
}
alias TypeInfo_Struct ТипСтрукт;
////////////////////////////////////

extern (D) class TypeInfo_Tuple : ИнфОТипе
{
    extern(C) extern ИнфОТипе[] elements;    alias elements элементы;
    ИнфОТипе[] getSetElements(ИнфОТипе[] elements = null);
    ИнфОТипе[] дайУстЭлементы(ИнфОТипе[] элементы = пусто);

    override ткст toString();
        override ткст вТкст();

    override int opEquals(Object o);

    hash_t getHash(ук p);
        т_хэш дайХэш(ук p);

    int equals(ук p1, ук p2);
        цел равны(ук p1, ук p2);

    int compare(ук p1, ук p2);
        цел сравни(ук p1, ук p2);

    override size_t tsize();
        override т_мера тразм();

    override проц swap(ук p1, ук p2);
        override проц поменяй( ук p1, ук p2);
}
alias TypeInfo_Tuple ТипКортеж;
////////////////////////////////////

extern (D) class TypeInfo_Const : ИнфОТипе
{
    override ткст toString() ;
        override ткст вТкст();

    override int opEquals(Object o);

    hash_t getHash(ук p);
        т_хэш дайХэш(ук p);
    int equals(ук p1, ук p2) ;
        цел равны(ук p1, ук p2);
    int compare(ук p1, ук p2) ;
        цел сравни(ук p1, ук p2);
    override size_t tsize() ;
        override т_мера тразм();
    override проц swap(ук p1, ук p2) ;
        override проц поменяй( ук p1, ук p2);

    override ИнфОТипе next() ;
        override ИнфОТипе следщ();
    override бцел flags() ;
        override бцел флаги();
    override проц[] init();
        override проц[] иниц();

    extern(C) extern ИнфОТипе base;    alias base основа;
    ИнфОТипе getSetBase(ИнфОТипе base = null);
    ИнфОТипе дайУстОву(ИнфОТипе основа = пусто);
}

```

```

alias TypeInfo_Const ТипКонстанта;
////////////////////////////////////////

extern (D) class TypeInfo_Invariant : TypeInfo_Const
{

    override ткст toString();
        override ткст вТкст();
}
alias TypeInfo_Invariant ТипИнвариант;
////////////////////////////////////////

// Object[]
extern (D) class TypeInfo_AC : TypeInfo_Array
{
    override т_хэш getHash(in ук p);
        override т_хэш дайХэш(in ук п);
    override цел equals(in ук p1, in ук p2);
        override цел равны(in ук p1, in ук p2);
    override цел compare(in ук p1, in ук p2);
        override цел сравни(in ук p1, in ук p2);
    override т_мера tsize();
        override т_мера тразм();
    override бцел flags();
        override бцел флаги();
    override ИнфоТипе next();
        override ИнфоТипе следщ();
}
alias TypeInfo_AC ТипОбъмас;
////////////////////////////////////////
// кдво[]
extern (D) class TypeInfo_Ar : TypeInfo_Array
{
    override ткст toString();
        override ткст вТкст();
    override т_хэш getHash(in ук p);
        override т_хэш дайХэш(in ук п);
    override цел equals(in ук p1, in ук p2);
        override цел равны(in ук p1, in ук p2);
    override цел compare(in ук p1, in ук p2);
        override цел сравни(in ук p1, in ук p2);
    override т_мера tsize();
        override т_мера тразм();
    override бцел flags();
        override бцел флаги();
    override ИнфоТипе next();
        override ИнфоТипе следщ();
}
alias TypeInfo_Ar ТипКдвомас;
////////////////////////////////////////

// кплав[]
extern (D) class TypeInfo_Aq : TypeInfo_Array
{

    override ткст toString();
        override ткст вТкст();
    override т_хэш getHash(in ук p) ;
        override т_хэш дайХэш(in ук п);
    override цел equals(in ук p1, in ук p2);
        override цел равны(in ук p1, in ук p2);
    override цел compare(in ук p1, in ук p2);
        override цел сравни(in ук p1, in ук p2);
    override т_мера tsize();
        override т_мера тразм();
    override бцел flags();
        override бцел флаги();
    override ИнфоТипе next();
        override ИнфоТипе следщ();
}
alias TypeInfo_Aq ТипКплавмас;
////////////////////////////////////////

// креал[]
extern (D) class TypeInfo_Ac : TypeInfo_Array
{

    override ткст toString();
        override ткст вТкст();
    override т_хэш getHash(in ук p) ;
        override т_хэш дайХэш(in ук п);
    override цел equals(in ук p1, in ук p2);
        override цел равны(in ук p1, in ук p2);
    override цел compare(in ук p1, in ук p2);
        override цел сравни(in ук p1, in ук p2);
    override т_мера tsize();
        override т_мера тразм();
    override бцел flags();
        override бцел флаги();
    override ИнфоТипе next();
}

```

```

        override ИнфоТипе следщ();
    }
    alias TypeInfo_Ac ТипКреалмас;
    //////////////////////////////////////

    // дво[]
    extern (D) class TypeInfo_Ad : TypeInfo_Array
    {

        override ткст toString();
        override ткст вТкст();
        override т_хэш getHash(in ук p) ;
        override т_хэш дайХэш(in ук p);
        override цел equals(in ук p1, in ук p2);
        override цел равны(in ук p1, in ук p2);
        override цел compare(in ук p1, in ук p2);
        override цел сравни(in ук p1, in ук p2);
        override т_мера tsize();
        override т_мера тразм();
        override бцел flags();
        override бцел флаги();
        override ИнфоТипе next();
        override ИнфоТипе следщ();
    }
    alias TypeInfo_Ad ТипДвомас;
    //////////////////////////////////////

    // вдво[]
    extern (D) class TypeInfo_Ap : TypeInfo_Ad
    {

        ткст toString();
        override ткст вТкст();
        override ИнфоТипе next();
        override ИнфоТипе следщ();
    }
    alias TypeInfo_Ap ТипВдвомас;
    //////////////////////////////////////

    // плав[]
    extern (D) class TypeInfo_Af : TypeInfo_Array
    {

        override ткст toString();
        override ткст вТкст();
        override т_хэш getHash(in ук p);
        override т_хэш дайХэш(in ук p);
        override цел equals(in ук p1, in ук p2);
        override цел равны(in ук p1, in ук p2);
        override цел compare(in ук p1, in ук p2);
        override цел сравни(in ук p1, in ук p2);
        override т_мера tsize();
        override т_мера тразм();
        override бцел flags();
        override бцел флаги();
        override ИнфоТипе next();
        override ИнфоТипе следщ();
    }
    alias TypeInfo_Af ТипПлавмас;
    //////////////////////////////////////

    // вплав[]
    extern (D) class TypeInfo_Ao : TypeInfo_Af
    {
        override ткст toString();
        override ткст вТкст();
        override ИнфоТипе next();
        override ИнфоТипе следщ();
    }
    alias TypeInfo_Ao ТипВплавмас;
    //////////////////////////////////////

    // байт[]
    extern (D) class TypeInfo_Ag : TypeInfo_Array
    {

        override ткст toString();
        override ткст вТкст();
        override т_хэш getHash(in ук p) ;
        override т_хэш дайХэш(in ук p);
        override цел equals(in ук p1, in ук p2);
        override цел равны(in ук p1, in ук p2);
        override цел compare(in ук p1, in ук p2);
        override цел сравни(in ук p1, in ук p2);
        override т_мера tsize();
        override т_мера тразм();
        override бцел flags();
        override бцел флаги();
        override ИнфоТипе next();
        override ИнфоТипе следщ();
    }

```

```

}
alias TypeInfo_Ag ТипБайтмас;
////////////////////////////////////

// байт[]
extern (D) class TypeInfo_Ah : TypeInfo_Ag
{
    override ткст toString();
        override ткст вТкст();
    override цел compare(in ук p1, in ук p2);
        override цел сравни(in ук p1, in ук p2);
    override ИнфОТипе next();
        override ИнфОТипе следщ();
}
alias TypeInfo_Ah ТипБбайтмас;
////////////////////////////////////

// проц[]
extern (D) class TypeInfo_Av : TypeInfo_Ah
{
    override ткст toString();
        override ткст вТкст();
    override ИнфОТипе next();
        override ИнфОТипе следщ();
}
alias TypeInfo_Av ТипПроцмас;
////////////////////////////////////

// bool[]
extern (D) class TypeInfo_Ab : TypeInfo_Ah
{
    override ткст toString();
        override ткст вТкст();
    override ИнфОТипе next();
        override ИнфОТипе следщ();
}
alias TypeInfo_Ab ТипБулмас;
////////////////////////////////////

// ткст
extern (D) class TypeInfo_Aa : TypeInfo_Ag
{
    override ткст toString();
        override ткст вТкст();
    override т_хэш getHash(in ук p);
        override т_хэш дайХэш(in ук п);
    override ИнфОТипе next();
        override ИнфОТипе следщ();
}
alias TypeInfo_Aa ТипТкст;
////////////////////////////////////

// цел[]
extern (D) class TypeInfo_Ai : TypeInfo_Array
{
    override ткст toString();
        override ткст вТкст();
    override т_хэш getHash(in ук p);
    override цел equals(in ук p1, in ук p2);
        override цел равны(in ук p1, in ук p2);
    override цел compare(in ук p1, in ук p2);
        override цел сравни(in ук p1, in ук p2);
    override т_мера tsize();
        override т_мера тразм();
    override бцел flags();
        override бцел флаги();
    override ИнфОТипе next();
        override ИнфОТипе следщ();
}
alias TypeInfo_Ai ТипЦелмас;
////////////////////////////////////

// бцел[]
extern (D) class TypeInfo_Ak : TypeInfo_Ai
{
    override ткст toString();
        override ткст вТкст();
    override цел compare(in ук p1, in ук p2);
        override цел сравни(in ук p1, in ук p2);
    override ИнфОТипе next();
        override ИнфОТипе следщ();
}
alias TypeInfo_Ak ТипБцелмас;
////////////////////////////////////
// юткст, дим[]
extern (D) class TypeInfo_Aw : TypeInfo_Ak
{
    override ткст toString();
        override ткст вТкст();
}

```

```

override ИнфоТипе next();
    override ИнфоТипе следщ();
}
alias TypeInfo_Aw ТипЮткст;
////////////////////////////////////

// дол[]
extern (D) class TypeInfo_Al : TypeInfo_Array
{
    override ткст toString();
    override ткст вТкст();
    override т_хэш getHash(in ук p);
    override цел equals(in ук p1, in ук p2);
    override цел равны(in ук p1, in ук p2);
    override цел compare(in ук p1, in ук p2);
    override цел сравни(in ук p1, in ук p2);
    override т_мера tsize();
    override т_мера тразм();
    override бцел flags();
    override бцел флаги();
    override ИнфоТипе next();
    override ИнфоТипе следщ();
}
alias TypeInfo_Al ТипДолмас;
////////////////////////////////////

// бдол[]
extern (D) class TypeInfo_Am : TypeInfo_Al
{
    override ткст toString();
    override ткст вТкст();
    override цел compare(in ук p1, in ук p2);
    override цел сравни(in ук p1, in ук p2);
    override ИнфоТипе next();
    override ИнфоТипе следщ();
}
alias TypeInfo_Am ТипБдолмас;
////////////////////////////////////

// реал[]
extern (D) class TypeInfo_Ae : TypeInfo_Array
{
    override ткст toString();
    override ткст вТкст();
    override т_хэш getHash(in ук p);
    override т_хэш дайХэш(in ук п);
    override цел equals(in ук p1, in ук p2);
    override цел равны(in ук p1, in ук p2);
    override цел compare(in ук p1, in ук p2);
    override цел сравни(in ук p1, in ук p2);
    override т_мера tsize();
    override т_мера тразм();
    override бцел flags();
    override бцел флаги();
    override ИнфоТипе next();
    override ИнфоТипе следщ();
}
alias TypeInfo_Ae ТипРеалмас;
////////////////////////////////////

// вреал[]
extern (D) class TypeInfo_Aj : TypeInfo_Ae
{
    override ткст toString();
    override ткст вТкст();
    override ИнфоТипе next();
    override ИнфоТипе следщ();
}
alias TypeInfo_Aj ТипВреалмас;
////////////////////////////////////

// крат[]
extern (D) class TypeInfo_As : TypeInfo_Array
{
    override ткст toString();
    override ткст вТкст();
    override т_хэш getHash(in ук p);
    override т_хэш дайХэш(in ук п);
    override цел equals(in ук p1, in ук p2);
    override цел равны(in ук p1, in ук p2);
    override цел compare(in ук p1, in ук p2);
    override цел сравни(in ук p1, in ук p2);
    override т_мера tsize();
    override т_мера тразм();
    override бцел flags();
    override бцел флаги();
    override ИнфоТипе next();
    override ИнфоТипе следщ();
}
alias TypeInfo_As ТипКратмас;

```

```

////////////////////////////////////

// Бкрат[]
extern (D) class TypeInfo_At : TypeInfo_As
{
    override ткст toString();
    override ткст вТкст();
    override цел compare(in ук p1, in ук p2);
    override цел сравни(in ук p1, in ук p2);
    override ИнфОТипе next();
    override ИнфОТипе следщ();
}
alias TypeInfo_At ТипБкратмас;
////////////////////////////////////

// шткст, шим[]
extern (D) class TypeInfo_Au : TypeInfo_At
{
    override ткст toString();
    override ткст вТкст();
    override ИнфОТипе next();
    override ИнфОТипе следщ();
}
alias TypeInfo_Au ТипШткст;
////////////////////////////////////

// байт
extern (D) class TypeInfo_g : ИнфОТипе
{
    override ткст toString();
    override ткст вТкст();
    override т_хэш getHash(in ук p);
    override т_хэш дайХэш(in ук п);
    override цел equals(in ук p1, in ук p2);
    override цел равны(in ук p1, in ук p2);
    override цел compare(in ук p1, in ук p2);
    override цел сравни(in ук p1, in ук p2);
    override т_мера tsize();
    override т_мера тразм();
    override проц swap(ук p1, ук p2);
    override проц поменяй( ук p1, ук p2);
}
alias TypeInfo_g ТипБайт;
////////////////////////////////////

// Объект
extern (D) class TypeInfo_C : ИнфОТипе
{
    override т_хэш getHash(in ук p);
    override т_хэш дайХэш(in ук п);
    override цел equals(in ук p1, in ук p2);
    override цел равны(in ук p1, in ук p2);
    override цел compare(in ук p1, in ук p2);
    override цел сравни(in ук p1, in ук p2);
    override т_мера tsize();
    override т_мера тразм();
    override бцел flags();
    override бцел флаги();
}
alias TypeInfo_C ТипОбъ;
////////////////////////////////////

// кдво
extern (D) class TypeInfo_r : ИнфОТипе
{
    override ткст toString();
    override ткст вТкст();
    override т_хэш getHash(in ук p);
    override т_хэш дайХэш(in ук п);
    override цел equals(in ук p1, in ук p2);
    override цел равны(in ук p1, in ук p2);
    override цел compare(in ук p1, in ук p2);
    override цел сравни(in ук p1, in ук p2);
    override т_мера tsize();
    override т_мера тразм();
    override проц swap(ук p1, ук p2);
    override проц поменяй( ук p1, ук p2);
    override проц[] init();
    override проц[] иниц();
}
alias TypeInfo_r ТипКдво;
////////////////////////////////////

// кплав
extern (D) class TypeInfo_q : ИнфОТипе
{
    override ткст toString();
    override ткст вТкст();
    override т_хэш getHash(in ук p);
    override т_хэш дайХэш(in ук п);
    override цел equals(in ук p1, in ук p2);
    override цел равны(in ук p1, in ук p2);
}

```

```

override цел compare(in ук p1, in ук p2);
    override цел сравни(in ук p1, in ук p2);
override т_мера tsize();
    override т_мера тразм();
override проц swap(ук p1, ук p2);
    override проц поменяй( ук p1, ук p2);
override проц[] init();
    override проц[] иниц();
}
alias TypeInfo_q ТипКплав;
////////////////////////////////////

//сим
extern (D) class TypeInfo_a : ИнфоТипе
{

override текст toString();
    override текст вТекст();
override т_хэш getHash(in ук p);
    override т_хэш дайХэш(in ук p);
override цел equals(in ук p1, in ук p2);
    override цел равны(in ук p1, in ук p2);
override цел compare(in ук p1, in ук p2);
    override цел сравни(in ук p1, in ук p2);
override т_мера tsize();
    override т_мера тразм();
override проц swap(ук p1, ук p2);
    override проц поменяй( ук p1, ук p2);
override проц[] init();
    override проц[] иниц();
}
alias TypeInfo_a ТипСим;
////////////////////////////////////

// креал
extern (D) class TypeInfo_c : ИнфоТипе
{

override текст toString();
    override текст вТекст();
override т_хэш getHash(in ук p);
    override т_хэш дайХэш(in ук p);
override цел equals(in ук p1, in ук p2);
    override цел равны(in ук p1, in ук p2);
override цел compare(in ук p1, in ук p2);
    override цел сравни(in ук p1, in ук p2);
override т_мера tsize();
    override т_мера тразм();
override проц swap(ук p1, ук p2);
    override проц поменяй( ук p1, ук p2);
override проц[] init();
    override проц[] иниц();
}
alias TypeInfo_c ТипКреал;
////////////////////////////////////

// дим
extern (D) class TypeInfo_w : ИнфоТипе
{

override текст toString();
    override текст вТекст();
override т_хэш getHash(in ук p);
    override т_хэш дайХэш(in ук p);
override цел equals(in ук p1, in ук p2);
    override цел равны(in ук p1, in ук p2);
override цел compare(in ук p1, in ук p2);
    override цел сравни(in ук p1, in ук p2);
override т_мера tsize();
    override т_мера тразм();
override проц swap(ук p1, ук p2);
    override проц поменяй( ук p1, ук p2);
override проц[] init();
    override проц[] иниц();
}
alias TypeInfo_w ТипДим;
////////////////////////////////////

// delegate
alias проц delegate(цел) дг;

extern (D) class TypeInfo_D : ИнфоТипе
{

override т_хэш getHash(in ук p);
    override т_хэш дайХэш(in ук p);
override цел equals(in ук p1, in ук p2);
    override цел равны(in ук p1, in ук p2);
override т_мера tsize();
    override т_мера тразм();
}

```



```

override проц swap(ук p1, ук p2);
    override проц поменяй( ук p1, ук p2);
override бцел flags();
    override бцел флаги();
}
alias TypeInfo_D ТипДг;
////////////////////

// дво
extern (D) class TypeInfo_d : ИнфОТипе
{
    override ткст toString() ;
        override ткст вТкст();
    override т_хэш getHash(in ук p);
        override т_хэш дайХэш(in ук p);
    override цел equals(in ук p1, in ук p2);
        override цел равны(in ук p1, in ук p2);
    override цел compare(in ук p1, in ук p2);
        override цел сравни(in ук p1, in ук p2);
    override т_мера tsize();
        override т_мера тразм();
    override проц swap(ук p1, ук p2);
        override проц поменяй( ук p1, ук p2);
    override проц[] init();
        override проц[] иниц();
}
alias TypeInfo_d ТипДво;
////////////////////

// плав
extern (D) class TypeInfo_f : ИнфОТипе
{

    override ткст toString() ;
        override ткст вТкст();
    override т_хэш getHash(in ук p);
        override т_хэш дайХэш(in ук p);
    override цел equals(in ук p1, in ук p2);
        override цел равны(in ук p1, in ук p2);
    override цел compare(in ук p1, in ук p2);
        override цел сравни(in ук p1, in ук p2);
    override т_мера tsize();
        override т_мера тразм();
    override проц swap(ук p1, ук p2);
        override проц поменяй( ук p1, ук p2);
    override проц[] init();
        override проц[] иниц();
}
alias TypeInfo_f ТипПлав;
////////////////////

// вдво
extern (D) class TypeInfo_p : TypeInfo_d
{

    override ткст toString();
        override ткст вТкст();
}
alias TypeInfo_p ТипВдво;
////////////////////
// вплав
extern (D) class TypeInfo_o : TypeInfo_f
{

    override ткст toString();
        override ткст вТкст();
}
alias TypeInfo_o ТипВплав;
////////////////////
// цел
extern (D) class TypeInfo_i : ИнфОТипе
{

    override ткст toString();
        override ткст вТкст();
    override т_хэш getHash(in ук p);
        override т_хэш дайХэш(in ук p);
    override цел equals(in ук p1, in ук p2);
        override цел равны(in ук p1, in ук p2);
    override цел compare(in ук p1, in ук p2);
        override цел сравни(in ук p1, in ук p2);
    override т_мера tsize();
        override т_мера тразм();
    override проц swap(ук p1, ук p2);
        override проц поменяй( ук p1, ук p2);
}
alias TypeInfo_i ТипЦел;
////////////////////
// вреал
extern (D) class TypeInfo_j : TypeInfo_e

```

```

{

override ткст toString();
    override ткст вТкст();
}
alias TypeInfo_j ТипВреал;
//////////
// дол
extern (D) class TypeInfo_l : ИнфоТипе
{

override ткст toString() ;
    override ткст вТкст();
override т_хэш getHash(in ук p);
    override т_хэш дайХэш(in ук п);
override цел equals(in ук p1, in ук p2);
    override цел равны(in ук p1, in ук p2);
override цел compare(in ук p1, in ук p2);
    override цел сравни(in ук p1, in ук p2);
override т_мера tsize();
    override т_мера тразм();
override проц swap(ук p1, ук p2);
    override проц поменяй( ук p1, ук p2);
}
alias TypeInfo_l ТипДол;
//////////

// указатель
extern (D) class TypeInfo_P : ИнфоТипе
{

override т_хэш getHash(in ук p);
    override т_хэш дайХэш(in ук п);
override цел equals(in ук p1, in ук p2);
    override цел равны(in ук p1, in ук p2);
override цел compare(in ук p1, in ук p2);
    override цел сравни(in ук p1, in ук p2);
override т_мера tsize();
    override т_мера тразм();
override проц swap(ук p1, ук p2);
    override проц поменяй( ук p1, ук p2);
override бцел flags();
    override бцел флаги();
}
alias TypeInfo_P ТипУк;
//////////

// реал
extern (D) class TypeInfo_e : ИнфоТипе
{

override ткст toString();
    override ткст вТкст();
override т_хэш getHash(in ук p);
    override т_хэш дайХэш(in ук п);
override цел equals(in ук p1, in ук p2);
    override цел равны(in ук p1, in ук p2);
override цел compare(in ук p1, in ук p2);
    override цел сравни(in ук p1, in ук p2);
override т_мера tsize();
    override т_мера тразм();
override проц swap(ук p1, ук p2);
    override проц поменяй( ук p1, ук p2);
override проц[] init();
    override проц[] иниц();
}
alias TypeInfo_e ТипРеал;
//////////

// крат
extern (D) class TypeInfo_s : ИнфоТипе
{

override ткст toString();
    override ткст вТкст();
override т_хэш getHash(in ук p);
    override т_хэш дайХэш(in ук п);
override цел equals(in ук p1, in ук p2);
    override цел равны(in ук p1, in ук p2);
override цел compare(in ук p1, in ук p2);
    override цел сравни(in ук p1, in ук p2);
override т_мера tsize();
    override т_мера тразм();
override проц swap(ук p1, ук p2);
    override проц поменяй( ук p1, ук p2);
}
alias TypeInfo_s ТипКрат;
//////////

// байт

```

```

extern (D) class TypeInfo_h : ИнфоТипе
{
    override ткст toString() ;
        override ткст вТкст();
    override т_хэш getHash(in ук p);
        override т_хэш дайХэш(in ук p);
    override цел equals(in ук p1, in ук p2);
        override цел равны(in ук p1, in ук p2);
    override цел compare(in ук p1, in ук p2);
        override цел сравни(in ук p1, in ук p2);
    override т_мера tsize();
        override т_мера тразм();
    override проц swap(ук p1, ук p2);
        override проц поменяй( ук p1, ук p2);
}
alias TypeInfo_h ТипВбайт;
////////////////////////////////////

extern (D) class TypeInfo_b : TypeInfo_h
{
    override ткст toString() ;
        override ткст вТкст();
}
alias TypeInfo_b ТипВул;
////////////////////////////////////

// бцел
extern (D) class TypeInfo_k : ИнфоТипе
{
    override ткст toString() ;
        override ткст вТкст();
    override т_хэш getHash(in ук p);
        override т_хэш дайХэш(in ук p);
    override цел equals(in ук p1, in ук p2);
        override цел равны(in ук p1, in ук p2);
    override цел compare(in ук p1, in ук p2);
        override цел сравни(in ук p1, in ук p2);
    override т_мера tsize();
        override т_мера тразм();
    override проц swap(ук p1, ук p2);
        override проц поменяй( ук p1, ук p2);
}
alias TypeInfo_k ТипВцел;
////////////////////////////////////

// бдол
extern (D) class TypeInfo_m : ИнфоТипе
{
    override ткст toString();
        override ткст вТкст();
    override т_хэш getHash(in ук p);
        override т_хэш дайХэш(in ук p);
    override цел equals(in ук p1, in ук p2);
        override цел равны(in ук p1, in ук p2);
    override цел compare(in ук p1, in ук p2);
        override цел сравни(in ук p1, in ук p2);
    override т_мера tsize();
        override т_мера тразм();
    override проц swap(ук p1, ук p2);
        override проц поменяй( ук p1, ук p2);
}
alias TypeInfo_m ТипВдол;
////////////////////////////////////

//бкрат
extern (D) class TypeInfo_t : ИнфоТипе
{
    override ткст toString();
        override ткст вТкст();
    override т_хэш getHash(in ук p);
        override т_хэш дайХэш(in ук p);
    override цел equals(in ук p1, in ук p2);
        override цел равны(in ук p1, in ук p2);
    override цел compare(in ук p1, in ук p2);
        override цел сравни(in ук p1, in ук p2);
    override т_мера tsize();
        override т_мера тразм();
    override проц swap(ук p1, ук p2);
        override проц поменяй( ук p1, ук p2);
}
alias TypeInfo_t ТипБкрат;
////////////////////////////////////

// проц
extern (D) class TypeInfo_v : ИнфоТипе
{

```

```

override текст toString();
    override текст вТекст();
override т_хэш getHash(in ук p);
    override т_хэш дайХэш(in ук p);
override цел equals(in ук p1, in ук p2);
    override цел равны(in ук p1, in ук p2);
override цел compare(in ук p1, in ук p2);
    override цел сравни(in ук p1, in ук p2);
override т_мера tsize();
    override т_мера тразм();
override проц swar(ук p1, ук p2);
    override проц поменяй( ук p1, ук p2);
override бцел flags();
    override бцел флаги();
}
alias TypeInfo_v ТипПроц;
////////////////////////////////////

//шим
extern (D) class TypeInfo_u : ИнфоТипе
{

override текст toString();
    override текст вТекст();
override т_хэш getHash(in ук p);
    override т_хэш дайХэш(in ук p);
override цел equals(in ук p1, in ук p2);
    override цел равны(in ук p1, in ук p2);
override цел compare(in ук p1, in ук p2);
    override цел сравни(in ук p1, in ук p2);
override т_мера tsize();
    override т_мера тразм();
override проц swar(ук p1, ук p2);
    override проц поменяй( ук p1, ук p2);
override проц[] init();
    override проц[] иниц();
}
alias TypeInfo_u ТипШим;
////////////////////////////////////

```

**Конец листинга модуля object \*\*\*\*\***

Внимательно ещё раз анализируя модуль object, можно прийти к следующему обобщению. В нём отчётливо понятно, что здесь заложена некоторая база для начала последующей рефлексии (Reflection), которая существует ныне в таких языках как Java и С#. Ну, и в других языках, конечно же тоже, так или иначе, используется.

Строго не ругайтесь за то, что несколько страниц "загрязнено" этим листингом! Ведь речь именно о том, что для справочной цели его можно разложить на две формы, два столбца таблицы. В одной части будет русский вариант, в другой – английский. Что и было показано на примере для класса Объект (Object).

Первая половина таблицы показывает 8 особых - комплексных - элементов языка. А начиная с девятого появляется ряд наследующих элементов от класса ИнформацияОТипе (ИнфоТипе).

```

struct СтруктураТипов
{

```

Объект	объ; //Object	Объект	Object
Исключение	искл; //Exception	Искл	Exception
Ошибка	ош; //Error	Ош	Error
Монитор	монитр; //Monitor	Монитор	Monitor
ИнфоМодуле	инфомод; //ModuleInfo	ИнфоМодуле	ModuleInfo
Интерфейс	ифейс; //Interface	Интерфейс	Interface
ИнфоКлассе	инфокласс; //ClassInfo	ИнфоКлассе	ClassInfo
ИнфоТипеИСмещ	инфотис; //OffsetTypeInfo	ИнфоТипеИСмещ	OffsetTypeInfo
*****	*****	*****	*****
ИнфоТипе	инфотип; //TypeInfo	ИнфоТипе	TypeInfo
ТипТипдеф	ттипдеф; //TypeInfo_Typedef	----X---	typedef
ТипПеречень	тперечнь; //TypeInfo_Enum	----X---	enum
ТипУказатель	туказ; //TypeInfo_Pointer: нет ли накладки с TypeInfo_P?	ук	void*
ТипМассив	тмас; //TypeInfo_Array	Массив[]	Array[]

ТипСтатМас	тстатмас; //TypeInfo_StaticArray	Массив[n]	Array[n]
ТипАссоцМас	тасоцмас; //TypeInfo_AssociativeArray	Массив[x][y]	Array[x][y]
ТипФункция	тфункц; //TypeInfo_Function	----X---	function
ТипДелегат	тделегат; //TypeInfo_Delegate	----X---	delegate
ТипКласс	ткласс; //TypeInfo_Class	----X---	class
ТипИнтерфейс	тифейс; //TypeInfo_Interface	----X---	interface
ТипСтрукт	тструкт; //TypeInfo_Struct	----X---	struct
ТипКортеж	ткортеж; //TypeInfo_Tuple	----X---	tuple
ТипКонстанта	тконстант; //TypeInfo_Const	----X---	const
ТипИнвариант	тинвар; //TypeInfo_Invariant	----X---	invariant
ТипОбъмас	тобъмас; //TypeInfo_AC	Объект[]	Object[]
ТипКдвомас	ткдвомс; //TypeInfo_Ar	кдво[]	cdouble[]
ТипКплавмас	ткплавмс; //TypeInfo_Aq	кплав[]	cfloat[]
ТипКреалмас	ткреалмс; //TypeInfo_Ac	креал[]	creal[]
ТипДвомас	тдвомс; //TypeInfo_Ad	дво[]	double[]
ТипВдвомас	твдвомс; //TypeInfo_Ap	вдво[]	idouble[]
ТипПлавмас	тплавмс; //TypeInfo_Af	плав[]	float[]
ТипВплавмас	твплавмс; //TypeInfo_Ao	вплав[]	ifloat[]
ТипБайтмас	тбайтмс; //TypeInfo_Ag	байт[]	byte[]
ТипБбайтмас	тббайтмс; //TypeInfo_Ah	ббайт[]	ubyte[]
ТипПроцмас	тпроцмс; //TypeInfo_Av	проц[]	void[]
ТипБулмас	тбулмс; //TypeInfo_Ab	бул[]	bool[]
ТипТкст	тткст; //TypeInfo_Aa	текст, сим[]	char[], string
ТипЦелмас	тцелмс; //TypeInfo_Ai	цел[]	int[]
ТипБцелмас	тбцелмс; //TypeInfo_Ak	бцел[]	uint[]
ТипЮткст	тюткст; //TypeInfo_Aw	юткст, дим[]	dchar[], dstring
ТипДолмас	тдолмс; //TypeInfo_Al	дол[]	long[]
ТипБдолмас	тбдолмс; //TypeInfo_Am	бдол[]	ulong[]
ТипРеалмас	треалмс; //TypeInfo_Ae	реал[]	real[]
ТипВреалмас	твреалмс; //TypeInfo_Aj	вреал[]	ireal[]
ТипКратмас	ткратмс; //TypeInfo_As	крат[]	short[]
ТипБкратмас	тбкратмс; //TypeInfo_At	бкрат[]	ushort[]
ТипШтекст	тшткст; //TypeInfo_Au	шткст, шим[]	wchar[], wstring
ТипБайт	тбайт; //TypeInfo_g	байт	byte
ТипОбъ	тобъ; //TypeInfo_C	Объект	Object
ТипКдво	ткдво; //TypeInfo_r	кдво	cdouble
ТипКплав	ткплав; //TypeInfo_q	кплав	cfloat
ТипСим	тсим; //TypeInfo_a	сим	char
ТипКреал	ткреал; //TypeInfo_c	креал	creal
ТипДим	тдим; //TypeInfo_w	дим	dchar
ТипДг	тделег; //TypeInfo_D	----X---	delegate
ТипДво	тдво; //TypeInfo_d	дво	double
ТипПлав	тплав; //TypeInfo_f	плав	float
ТипВдво	твдво; //TypeInfo_p	вдво	idouble
ТипВплав	твплав; //TypeInfo_o	вплав	ifloat
ТипЦел	тцел; //TypeInfo_i	цел	int
ТипВреал	твреал; //TypeInfo_j	вреал	ireal
ТипДол	тдол; //TypeInfo_l	дол	long
ТипУк	тук; //TypeInfo_P: в чём его разница с TypeInfo_Pointer!?.&?	----X---	ref(&)
ТипРеал	треал; //TypeInfo_e	реал	real
ТипКрат	ткрат; //TypeInfo_s	крат	short
ТипБбайт	тббайт; //TypeInfo_h	ббайт	ubyte
ТипБул	тбул; //TypeInfo_b	бул	bool
ТипБцел	тбцел; //TypeInfo_k	бцел	uint
ТипБдол	тбдол; //TypeInfo_m	бдол	ulong

ТипБкрат	тбкрат; //TypeInfo_t	бкрат	ushort
ТипПроц	тпроц; //TypeInfo_v	проц	void
ТипШим	тшим; //TypeInfo_u	шим	wchar

}

Вот, собственно, и вся информация, которая вошла в один из основных модулей.

Знатоки D сразу обнаружат, что в D расположение данных модулей совершенно иное. По сути здесь собрано несколько отдельных модулей в одно целое.

Кроме того, наличие **extern(D)** говорит о том, что классы эти находятся в динамической библиотеке. И это действительно так, - вся основная - базовая - функциональность помещена в Dll.

d:\dinrus\dev\DIRUS\Base\*.*	
Имя	↑Тип
[..]	
[base]	
[import]	
[static]	
[TangoAdaptation]	
[Test]	
mk	bat
clean	d
csf	d
base	def
Dinrus.Base	dll
clean	exe
sc	ini
base	res

Судя по вырезке экрана, это Dinrus.Base.dll.

В папке **base** собрано всё, что входит в эту библиотеку.

Папка **import** содержит краткие интерфейсные сводки по импортам, которые будут использоваться далее для контакта компилятора с библиотекой.

В папке **static** находится другая часть, статическая. Она не входит в dll, но помещается в статическую библиотеку Dinrus.lib

d:\dinrus\lib\*.*			
Имя	↑Тип	Размер	Дата
[..]			
[c]			
[gdk-pixbuf-2.0]			
[gtk-2.0]			
[lulada]			
[lulada_eng]			
[sysimport]			
[tcl8.5]			
[tk8.5]			
Dinrus	lib	10 359 808	14.08.2016 07:54
Dinrus.CBase	lib	66 048	26.12.2015 06:03
DinrusArcDLL	lib	158 208	26.12.2015 07:59
DinrusBaseDLL	lib	968 192	14.08.2016 07:53
DinrusConc	lib	1 434 624	10.08.2016 22:19
DinrusDbi	lib	1 353 728	20.07.2016 07:26
DinrusMinidDLL	lib	1 536	26.12.2015 07:59
DinrusSpecBuild	lib	108 032	14.08.2016 07:55
DinrusTango	lib	7 627 264	13.08.2016 12:06
DinrusWin32	lib	8 729 088	14.08.2016 07:54
DinrusWinDLL	lib	15 360	20.07.2016 16:08
glu	lib	9 216	27.01.2015 22:22
glut	lib	20 992	27.01.2015 22:21
import	lib	5 786 624	27.01.2015 22:15
mesa	lib	200 192	11.10.2012 21:20
std	lib	1 403 392	03.02.2016 00:17
ucrt	lib	301 568	05.10.2015 19:50
minit	obj	319	17.10.2010 20:10

Вся работа по компиляции библиотек выполняется с помощью скрипта mk.bat. К сожалению, VisualD, который позволяет работать с языком D в Microsoft Visual Studio, на данный момент нацелен на работу с языком D v2. Поэтому придётся заняться разработкой собственной привязки к этой студии. Желательно, и к CLR. На данный момент ничто не препятствует тому, чтобы активно портировать Динрус на все системы.

Так, проект DSharp уже есть среди прочих на <http://github.com/DinrusGroup>. Осталось найти на всё время!...

## Листинга mk.bat для DinrusBase \*\*\*\*\*

```
:::This file was developed for Dinrus Programming Language by Vitaliy Kulich  
:::Copyright is by Dinrus Group.
```

```
:back  
:::Setting environment variables  
@set this=%DINRUS%\..\dev\DINRUS\Base  
@set R=%DINRUS%\..\imp\dinrus  
@set LIBS=%DINRUS%\..\lib\sysimport  
@set LDIR=%DINRUS%\..\lib  
@set DMD=%DINRUS%\dmd.exe  
@set DMC=%DINRUS%\dmc.exe  
@set LIB=%DINRUS%\lib.exe  
@set IMPLIB=%DINRUS%\implib.exe  
@set ARCDIR=%this%\..\Arc  
@set MINIDDIR=%this%\..\Minid  
@set LS=%DINRUS%\ls2.exe  
@set PACK=%DINRUS%\upx.exe  
  
::goto Lib  
  
:::Deleting previous objects  
@del %LDIR%\Dinrus.lib  
@del %LDIR%\Dinrus.bak  
@del %this%\*.rsp  
@del %this%\*.obj  
@del %this%\*.map  
@del %this%\*.dll  
@del %this%\base\rt\*.obj  
@del %this%\*.lib  
@del %this%\*.exe  
  
:::Files with staff that must be same in imports and base-making  
::: just copied to imports immediately? without manual processing  
  
:copy %this%\base\sys\DConsts.d %this%\import\sys\DConsts.d  
:copy %this%\base\sys\DStructs.d %this%\import\sys\DStructs.d  
:copy %this%\base\sys\DTypes.d %this%\import\sys\DTypes.d  
:copy %this%\base\sys\Diface.d %this%\import\sys\Diface.d  
:copy %this%\base\base.d %this%\import\base.d  
  
:::Making dirs for di files in \imp\dinrus\  
::: and copying imports from .\import folder to them  
  
mkdir %R%  
copy %this%\import\*.d %R%\*.di  
  
mkdir %R%\std  
copy %this%\import\std\*.d %R%\std\*.di  
  
mkdir %R%\tpl  
copy %this%\import\tpl\*.d %R%\tpl\*.di  
  
mkdir %R%\st  
copy %this%\import\st\*.d %R%\st\*.di  
  
mkdir %R%\mesh  
copy %this%\import\mesh\*.d %R%\mesh\*.di  
  
mkdir %R%\win32  
mkdir %R%\win32\directx  
copy %this%\..\win32\*.d %R%\win32\*.di  
copy %this%\..\win32\directx\*.d %R%\win32\directx\*.di  
  
mkdir %R%\def  
copy %this%\..\win32\directx\*.def %R%\def\*.def  
  
mkdir %R%\sys  
mkdir %R%\sys\inc  
mkdir %R%\sys\COM  
copy %this%\import\sys\*.d %R%\sys\*.di  
copy %this%\import\sys\inc\*.d %R%\sys\inc\*.di  
copy %this%\import\sys\COM\*.d %R%\sys\COM\*.di  
  
mkdir %R%\lib  
copy %this%\import\lib\*.d %R%\lib\*.di  
  
mkdir %R%\col  
mkdir %R%\col\model  
copy %this%\import\col\*.d %R%\col\*.di  
copy %this%\import\col\model\*.d %R%\col\model\*.di  
  
mkdir %R%\linalg  
copy %this%\import\linalg\*.d %R%\linalg\*.di  
  
mkdir %R%\geom  
copy %this%\import\geom\*.d %R%\geom\*.di
```

```

mkdir %R%\util
copy %this%\import\util\*.d %R%\util\*.di

::mkdir %R%\io
::mkdir %R%\io\device
::mkdir %R%\io\stream
::copy %this%\import\io\*.d %R%\io\*.di
::copy %this%\import\io\device\*.d %R%\io\*.di
::copy %this%\import\io\stream\*.d %R%\io\*.di

:::Compiling C code

%DMC% -c -o%this%\complex.obj %this%\base\rt\complex.c -I%DINRUS%\..\include
%DMC% -c -o%this%\critical.obj %this%\base\rt\critical.c -I%DINRUS%\..\include
%DMC% -c -o%this%\deh.obj %this%\base\rt\deh.c -I%DINRUS%\..\include
%DMC% -c -o%this%\monitor.obj %this%\base\rt\monitor.c -I%DINRUS%\..\include

%DMD% -lib -of%this%\Cdinr.lib %this%\complex.obj %this%\critical.obj %this%\deh.obj %
this%\monitor.obj

:Base
:::Creating respond file
::%this%\base\io\*.d %this%\base\io\device\*.d %this%\base\io\stream\*.d
::%LS% -d %this%\base\io\*.d %this%\base\io\device\*.d %this%\base\io\stream\*.d
%LS% -d %this%\base\std\*.d %this%\base\*.d %this%\base\tpl\*.d %this%\base\rt\*.d %this%
\base\sys\*.d %this%\base\sys\inc\*.d>>%this%\objs.rsp

:::Make Dinrus.Base.dll

@if exist %DINRUS%\dinrus.exe %DINRUS%\dinrus.exe

%DMD% -g -O -debug -of%this%\Dinrus.Base.dll %this%\static\dll.d @%this%\objs.rsp %this%
\base.def %this%\base.res %LDIR%\minit.obj %LDIR%\import.lib %this%\Cdinr.lib

@if not exist %this%\Dinrus.Base.dll pause
@if exist %this%\Dinrus.Base.dll goto nextStep
@del %this%\objs.rsp

@goto Base

:nextStep
:::Make its import lib
%IMPLIB% /system %this%\Dinrus.lib %this%\Dinrus.Base.dll
%IMPLIB% /system %this%\DinrusBaseDLL.lib %this%\Dinrus.Base.dll
copy %this%\DinrusBaseDLL.lib %LDIR%
::copy %this%\Dinrus.Base.dll %DINRUS%
::copy %this%\Dinrus.Base.dll c:\Windows\system32

:::To compress
:%PACK% %this%\Dinrus.Base.dll

:::Clean
@del %this%\*.obj

:::Compiling imports into static part of dinrus.lib

%DMD% -c -O -g -of%this%\cidrus.obj %this%\import\cidrus.d -I%R%
%DMD% -c -O -g -of%this%\stdrus.obj %this%\import\stdrus.d -I%R%
%DMD% -c -O -g -of%this%\runtime.obj %this%\import\runtime.d -I%R%
%DMD% -c -O -g -of%this%\object.obj %this%\import\object.d -I%R%
%DMD% -c -O -g -of%this%\gc.obj %this%\import\gc.d -I%R%
%DMD% -c -O -g -of%this%\thread.obj %this%\import\thread.d -I%R%
%DMD% -c -O -g -of%this%\sync.obj %this%\import\sync.d -I%R%
%DMD% -c -O -g -of%this%\import\tracer.d
%DMD% -c -O -g -of%this%\ini.obj %this%\static\ini.d -I%R%
%DMD% -c -O -g -of%this%\stringz.obj %this%\import\stringz.d -I%R%

%DMD% -c -O -g -of%this%\win.obj %this%\import\win.d -I%R%
%DMD% -c -O -g -of%this%\wincom.obj %this%\import\com.d -I%R%
%DMD% -c -O -g -of%this%\dinrus.obj %this%\import\dinrus.d -I%R%
%DMD% -c -O -g -of%this%\exception.obj %this%\import\exception.d -I%R%
:::%DMD% -c -O -g %this%\import\openrj.d

%DMD% -c -O -g -of%this%\rotozoom.obj %this%\static\rotozoom.d -I%R%
%DMD% -c -O -g -of%this%\msscript.obj %this%\static\msscript.d DRwin32.lib -I%R%
%DMD% -c -O -g -of%this%\activex.obj %this%\static\activex.d DRwin32.lib -I%R%
%DMD% -c -O -g -of%this%\json.obj %this%\static\json.d -I%R%

:::Special configuration items
%DMD% -c -O -g -of%this%\base.obj %this%\static\base.d -I%R%
%DMD% -c -O -g -of%this%\exeMain.obj %this%\static\exeMain.d -I%R%

:%DMD% -c -O -g exeef.d

%DMD% -c -O -g -of%this%\winapi.obj %this%\import\winapi.d -I%R%

```



```

%DMD% -c -O -g -of %this%\global.obj %this%\import\global.d -I %R%

%DMD% -c -O -g -of %this%\all.obj %this%\import\tpl\all.d -I %R%
%DMD% -c -O -g -of %this%\alloc.obj %this%\import\tpl\alloc.d -I %R%
%DMD% -c -O -g -of %this%\bind.obj %this%\import\tpl\bind.d -I %R%
%DMD% -c -O -g -of %this%\box.obj %this%\import\tpl\box.d -I %R%
%DMD% -c -O -g -of %this%\collection.obj %this%\import\tpl\collection.d -I %R%
%DMD% -c -O -g -of %this%\metastrings.obj %this%\import\tpl\metastrings.d -I %R%
%DMD% -c -O -g -of %this%\minmax.obj %this%\import\tpl\minmax.d -I %R%
%DMD% -c -O -g -of %this%\signal.obj %this%\import\tpl\signal.d -I %R%
%DMD% -c -O -g -of %this%\args.obj %this%\import\tpl\args.d -I %R%
%DMD% -c -O -g -of %this%\traits.obj %this%\import\tpl\traits.d -I %R%
%DMD% -c -O -g -of %this%\tupletuple.obj %this%\import\tpl\tupletuple.d -I %R%
%DMD% -c -O -g -of %this%\stream.obj %this%\import\tpl\stream.d -I %R%
%DMD% -c -O -g -of %this%\singleton.obj %this%\import\tpl\singleton.d -I %R%
%DMD% -c -O -g -of %this%\comtpl.obj %this%\import\tpl\com.d -I %R%
%DMD% -c -O -g -of %this%\std.obj %this%\import\tpl\std.d -I %R%
%DMD% -c -O -g -of %this%\weakref.obj %this%\import\tpl\weakref.d -I %R%

:pause
%DMD% -c -O -g -of %this%\DStructs.obj %this%\import\sys\DStructs.d -I %R%
%DMD% -c -O -g %this%\import\sys\registry.d
%DMD% -c -O -g -of %this%\DIfaces.obj %this%\import\sys\DIfaces.d -I %R%
%DMD% -c -O -g -of %this%\DConsts.obj %this%\import\sys\DConsts.d -I %R%
%DMD% -c -O -g -of %this%\DFuncs.obj %this%\import\sys\DFuncs.d -I %R%
%DMD% -c -O -g -of %this%\DProcess.obj %this%\import\sys\DProcess.d -I %R%

%DMD% -c -O -g -of %this%\kernel32.obj %this%\import\sys\inc\kernel32.d -I %R%

:%DMD% -c -O -g %this%\import\sys\en.d
%DMD% -c -O -g -of %this%\memory.obj %this%\import\sys\memory.d -I %R%
%DMD% -c -O -g -of %this%\uuid.obj %this%\import\sys\uuid.d -I %R%
%DMD% -c -O -g -of %this%\comsys.obj %this%\static\sys0\com.d -I %R%

%DMD% -c -O -g -of %this%\shell32.obj %this%\import\sys\COM\shell32.d -I %R%
%DMD% -c -O -g -of %this%\scomall.obj %this%\import\sys\COM\all.d -I %R%

:%DMD% -c -O -g %this%\static\lib\mesa.d mesa.lib

:%DMD% -c -O -g %this%\import\stddinrus\base64.d

:%DMD% -c -O -g -ofrt.obj @dobjjs.rsp

:::Making library with static content
:dinrus2

%DMD% -lib -of %this%\dinrus2.lib %this%\base.obj %this%\object.obj %this%\cidrus.obj %
this%\stdrus.obj %this%\dinrus.obj %this%\win.obj %this%\runtime.obj %this%\gc.obj %this%
\thread.obj %this%\sync.obj %this%\stringz.obj %this%\all.obj %this%\bind.obj %this%
\box.obj %this%\metastrings.obj %this%\minmax.obj %this%\signal.obj %this%\args.obj %this%
\tupletuple.obj %this%\traits.obj %this%\exception.obj %LDIR%\minit.obj %this%\DStructs.obj
%this%\DIfaces.obj %this%\DConsts.obj %this%\DFuncs.obj %this%\DProcess.obj %this%
\comtpl.obj %this%\wincom.obj %this%\shell32.obj %this%\stream.obj %this%\memory.obj %
this%\msscript.obj %this%\activex.obj %this%\winapi.obj %this%\singleton.obj %this%
\alloc.obj %this%\collection.obj %this%\kernel32.obj %this%\ini.obj %this%\Std.obj %this%
\exeMain.obj %this%\uuid.obj %this%\comsys.obj %this%\rotozoom.obj %this%\scomall.obj %
this%\global.obj %this%\weakref.obj %this%\registry.obj %this%\Cdinr.lib

@if exist %this%\dinrus2.lib goto Join
@if not exist %this%\dinrus2.lib pause
cls
@goto dinrus2
:::Adding static libraries to Dinrus.lib
:Join
%LIB% -p256 %this%\Dinrus.lib %this%\dinrus2.lib

:::Compiling codes from .\static folder

:Lib

%LS% -d %this%\import\lib\*.d >>%this%\lib.rsp
%DMD% -lib -of %this%\dlib.lib @%this%\lib.rsp
@if exist %this%\dlib.lib del %this%\lib.rsp
@if exist %this%\dlib.lib goto Col
@if not exist %this%\dlib.lib pause
@del %this%\col.rsp
cls
@goto Lib
pause

:Col
%LS% -d %this%\static\col\*.d %this%\static\col\model\*.d >>%this%\col.rsp
%DMD% -lib -of %this%\col.lib @%this%\col.rsp
@if exist %this%\col.lib del %this%\col.rsp
@if exist %this%\col.lib goto Util
@if not exist %this%\col.lib pause
@del %this%\col.rsp
cls
@goto Col

```

```

:Util
%LS% -d %this%\static\util\*.d>>%this%\ut.rsp
%DMD% -lib -of%this%\util.lib @%this%\ut.rsp
@if exist %this%\util.lib del %this%\ut.rsp
@if exist %this%\util.lib goto LinAlg
@if not exist %this%\util.lib pause
@del %this%\ut.rsp
cls
@goto Util

:LinAlg
%LS% -d %this%\static\linalg\*.d>>%this%\la.rsp
%DMD% -lib -of%this%\la.lib @%this%\la.rsp
@if exist %this%\la.lib del %this%\la.rsp
@if exist %this%\la.lib goto Mesh
@if not exist %this%\la.lib pause
@del %this%\la.rsp
cls
@goto LinAlg

:Mesh
%LS% -d %this%\static\mesh\*.d>>%this%\mesh.rsp
%DMD% -lib -of%this%\mesh.lib @%this%\mesh.rsp
@if exist %this%\mesh.lib del %this%\mesh.rsp
@if exist %this%\mesh.lib goto St
@if not exist %this%\mesh.lib pause
@del %this%\mesh.rsp
cls
@goto Mesh

:St
%LS% -d %this%\static\st\*.d>>%this%\st.rsp
%DMD% -lib -of%this%\st.lib @%this%\st.rsp
@if exist %this%\st.lib del %this%\st.rsp
@if exist %this%\st.lib goto Geom
@if not exist %this%\st.lib pause
@del %this%\st.rsp
cls
@goto St

:Geom
%LS% -d %this%\static\geom\*.d>>%this%\geom.rsp
%DMD% -lib -of%this%\geom.lib @%this%\geom.rsp
@if exist %this%\geom.lib del %this%\geom.rsp
@if exist %this%\geom.lib goto IO
@if not exist %this%\geom.lib pause
@del %this%\geom.rsp
cls
@goto Geom

:IO
:goto DRwin32
%LS% -d %this%\import\io\*.d %this%\import\io\device\*.d %this%\import\io\stream\*.d>>%
this%\io.rsp
%DMD% -lib -of%this%\io.lib @%this%\io.rsp
@if exist %this%\io.lib del %this%\io.rsp
@if exist %this%\io.lib goto DRwin32
@if not exist %this%\io.lib pause
@del %this%\io.rsp
cls
@goto IO

:DRwin32
:::Makin Dinrus_win32.lib
@if exist %LDIR%\DinrusWin32.lib goto skip
%LS% -d %this%\..\win32\*.d %this%\..\win32\directx\*.d %this%\..\win32\directx\*.def>>%
this%\win32.rsp
%DMD% -O -release -version=Unicode -lib -of%this%\DinrusWin32.lib @%this%\win32.rsp
if exist %this%\win32.rsp del %this%\win32.rsp
if not exist %this%\DinrusWin32.lib pause
copy %this%\DinrusWin32.lib /b %LDIR%\DinrusWin32.lib /b

:skip
:::goto finish

:Dinrus.Arc.dll
:::Making Dinrus.Arc.dll
cd %ARCDIR%
%DINRUS%\rulada
%DMD% -of%ARCDIR%\Dinrus.Arc.dll %ARCDIR%\dll.d %ARCDIR%\arc.d %ARCDIR%\arcus.def %ARCDIR%
\arcus.res derelict.lib arc.lib
%IMPLIB% /system %ARCDIR%\DinrusArcDLL.lib %ARCDIR%\Dinrus.Arc.dll
copy %ARCDIR%\DinrusArcDLL.lib %LDIR%
%PACK% %ARCDIR%\Dinrus.Arc.dll
copy %ARCDIR%\Dinrus.Arc.dll %DINRUS%
del %ARCDIR%\*.dll %ARCDIR%\*.obj %ARCDIR%\*.rsp %ARCDIR%\*.map

:Dinrus.Minid.dll
cd %MINIDDIR%

```

```

%LS% -d %MINIDDIR%\*.d >>%MINIDDIR%\objs.rsp
%DMD% -g -O -cov -of %MINIDDIR%\Dinrus.Minid.dll @%MINIDDIR%\objs.rsp %MINIDDIR%\minid.def
%MINIDDIR%\minid.res
%IMPLIB% /system %MINIDDIR%\DinrusMinidDLL.lib %MINIDDIR%\Dinrus.Minid.dll
%PACK% %MINIDDIR%\Dinrus.Minid.dll
copy %MINIDDIR%\DinrusMinidDLL.lib %LDIR%
copy %MINIDDIR%\Dinrus.Minid.dll %DINRUS%
del %MINIDDIR%\*.dll %MINIDDIR%\*.obj %MINIDDIR%\*.rsp %MINIDDIR%\*.map
cd %this%
%DINRUS%\dinrus

:finish

%LIB% -p256 %this%\Dinrus.lib %this%\dlib.lib
%LIB% -p256 %this%\Dinrus.lib %this%\col.lib
%LIB% -p256 %this%\Dinrus.lib %this%\util.lib
%LIB% -p256 %this%\Dinrus.lib %this%\la.lib
%LIB% -p256 %this%\Dinrus.lib %this%\geom.lib
%LIB% -p256 %this%\Dinrus.lib %this%\mesh.lib
%LIB% -p256 %this%\Dinrus.lib %this%\st.lib
::%LIB% -p256 %this%\Dinrus.lib %this%\io.lib
%LIB% -p256 %this%\Dinrus.lib %ARCDIR%\arc2.lib
%LIB% -p256 %this%\Dinrus.lib %MINIDDIR%\rminid.lib

:::Adding system imports
:::%LIB% -p256 Dinrus.lib %LDIR%\import.lib

:::Copying Dinrus.lib to main Dinrus lib folder
%LIB% -p256 Dinrus.lib %LDIR%\import.lib
copy %this%\Dinrus.lib %LDIR%
copy %this%\Dinrus.Base.dll %DINRUS%

%DMD% -lib -of %this%\DinrusSpecBuild.lib %this%\static\dllMain.d
copy %this%\DinrusSpecBuild.lib %LDIR%

del %this%\*.lib %this%\*.obj
:::Cleaning
%DMD% %this%\clean.d
%this%\clean
::: same with the Dll - to bin folder

::del *.lib *.dll
cd ..\Exe
mk.bat
exit

Конец листинга mk.bat для DinrusBase *****

```

О, сколько времени ушло на все эти вещи! Но теперь - на самом Динрусе - можно написать это всё гораздо быстрее и проще...)))

В общем, выкладываю здесь и содержание пускового файла, так как с ним пришлось изрядно потрудиться. Специалист должен сразу же почерпнуть массу информации о составе Dinrus.lib, а также о моём плохом владении командной оболочкой WindowsXP.

Да, именно этой ОС, т.к. для программирования она самая - на данный момент - "либеральная". Предпочитаю иметь Windows 10 или 7 (что более прагматично) в форме виртуалки, работающей через, например, вот такую программу:



Названия программ, которые я бы порекомендовал, как видите, постарался захватить в объектив "фотокамеры".

Да, кстати, Notepad++ - это единственная серьёзная программа, которая уделяет внимание правильности кодировки и даёт все возможности делать нужные настройки. Ряд других IDE, которые приходилось протестировать, успел сильно разочаровать в этом вопросе.

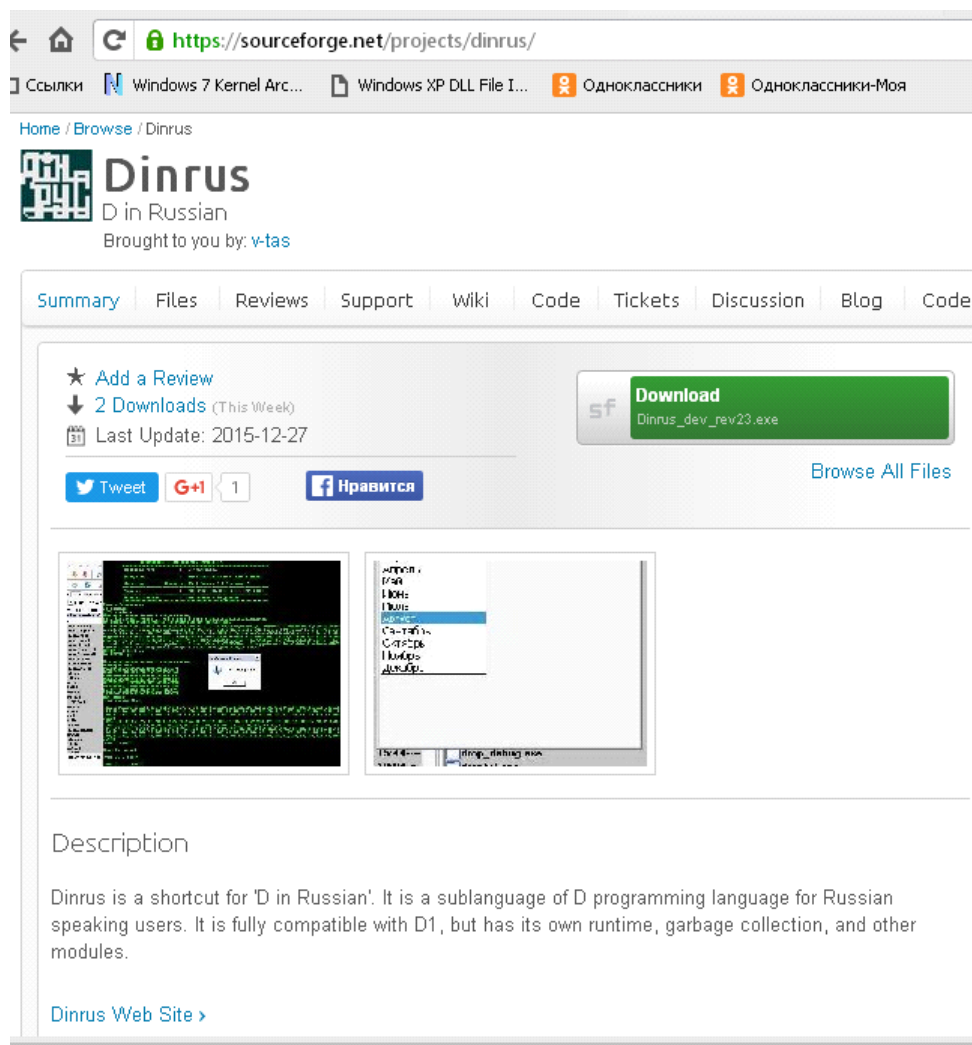
Теперь перейдём к практической части, касающейся разработки.

Первым делом, вам понадобится зайти по адресу <http://sf.net/p/dinrus>, для того чтобы скачать весь необходимый бинарно-файловый набор.

Если вы его опустите (после распаковки) на диск D:, в папку dinrus, то вам менее всего придётся беспокоиться о переменной среды DINRUS, установленной, - в моём случае, - в D://dinrus/bin

Именно от этой координаты пляшут все скрипты.

Кроме того, рекомендую также скачать проект через SVN-клиентскую программу, чтобы получить "полноту" информации.



Вот так выглядит то место, куда я вас направил!)))

[svn checkout svn://svn.code.sf.net/p/dinrus/code-0/dinrus-code-0](http://svn.code.sf.net/p/dinrus/code-0/dinrus-code-0)

С помощью такого кода вы сможете загрузить файлы проекта. Но помимо этого есть ещё другая страница, <http://sf.net/p/rulada>, откуда пошла ешь началась разработка. Рулада - это набор библиотек, слабо руссифицированный, почти англоязычный. Именно с ним придётся ХОРОШО ПОВОЗИТЬСЯ, чтобы БОГАТИТЬ ДИНРУС полноценным функционалом.

Сделаю скриншот в папке импорта Рулады, чтобы произвести впечатление...

d:\dinrus\lib\rulada_eng\*.*			
Имя	↑Тип	Размер	Дата
[..]		<DIR>	16.10.2015 22:18
amigos	lib	7 070 208	26.02.2015 18:11
arc	lib	14 430 720	26.02.2015 18:13
auxc	lib	5 208 064	26.02.2015 18:09
auxd	lib	4 298 240	26.02.2015 18:12
derelict	lib	6 301 696	26.02.2015 18:09
gtkD	lib	13 638 656	26.02.2015 18:41
kong	lib	343 552	26.02.2015 18:12
mango	lib	2 881 536	26.02.2015 18:12
rulada	lib	14 238 720	26.02.2015 18:06
tango	lib	5 354 496	26.02.2015 18:08
gcstub	obj	4 108	26.02.2015 18:05

Фактически в этот десяток библиотек мною всунута вся Windows с потрохами!!!!)

Это легко заметить по другому скриншоту, в котором ощущается только внешнее количество имеющихся пакетов!!!!)

d:\dinrus\imp\rulada_eng\*.*			
Имя	↑Тип	Размер	Дата
[amigos]		<DIR>	12.12.2014 1
[arc]		<DIR>	12.12.2014 1
[auxc]		<DIR>	12.12.2014 1
[auxd]		<DIR>	12.12.2014 1
[bcd]		<DIR>	25.01.2015 2
[dbi]		<DIR>	12.12.2014 1
[dcollections]		<DIR>	12.12.2014 1
[ddl]		<DIR>	26.01.2015 2
[derelict]		<DIR>	12.12.2014 1
[dgui]		<DIR>	06.01.2015 C
[gtkD]		<DIR>	12.12.2014 1
[hcf]		<DIR>	25.01.2015 2
[kong]		<DIR>	12.12.2014 1
[mango]		<DIR>	12.12.2014 1
[meta]		<DIR>	12.12.2014 1
[os]		<DIR>	20.01.2016 C
[rae]		<DIR>	12.12.2014 1
[rt]		<DIR>	12.12.2014 1
[rulada]		<DIR>	17.02.2015 C
[std]		<DIR>	18.02.2015 C
[std2]		<DIR>	12.12.2014 1
[sys]		<DIR>	20.02.2015 C
[tango]		<DIR>	12.12.2014 1
[tpl]		<DIR>	20.02.2015 C
[utils]		<DIR>	12.12.2014 1
cidrus	d	70 916	26.12.2015 C
dll	d	815	17.12.2013 1
crc32	di	3 832	19.12.2013 C
gcstats	di	1 502	17.12.2013 1
object	di	14 738	17.02.2015 C

К сожалению, не могу показать в деле, но возможности этого набора очень велики, и... пока ещё мало исхожены ногами программными его тропы-стежи)))...

Как видите, и Tango и Phobos здесь собраны в одном флаконе. Фактически, пришлось перекачать половину <http://dsourcе.org>, чтобы там обрести данные "богатства")))

Итак, подходим к заключительной части "Введения"... Многие "введения" столь часто так и остаются просто "введениями"... а здесь нужна усердная и серьёзная КОЛЛЕКТИВНАЯ работа... Именно поэтому...

Жду вас на <http://ok.ru/vit.klich>

Или на почтовом ящике [dinruspro@mail.ru](mailto:dinruspro@mail.ru)

Кроме того, предлагаю посетить и подключиться к работе окончательно, по адресу <http://github.com/DinrusGroup>

На этом желаю успехов в нелёгком предстоящем пути)))

P.S. Начну постепенно сочинять практические методички по дальнейшей совместной работе. Надеюсь, скоро UPP-Динрус, Dsharp, и Динрус-Android.. все эти чудеса фантазии воплотятся в нечто реальное!)))

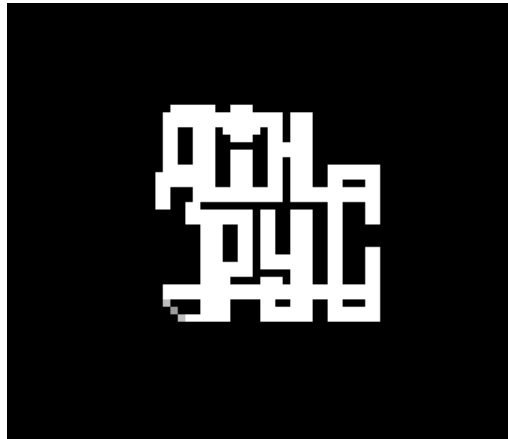
Ну, и "Привет, Мир" (на всякий случай):

**import dinrus;**

```

проц main()
{
    нс;
    таб;
    скажинс("Привет, Мир!");

```



```

    пауза;
}

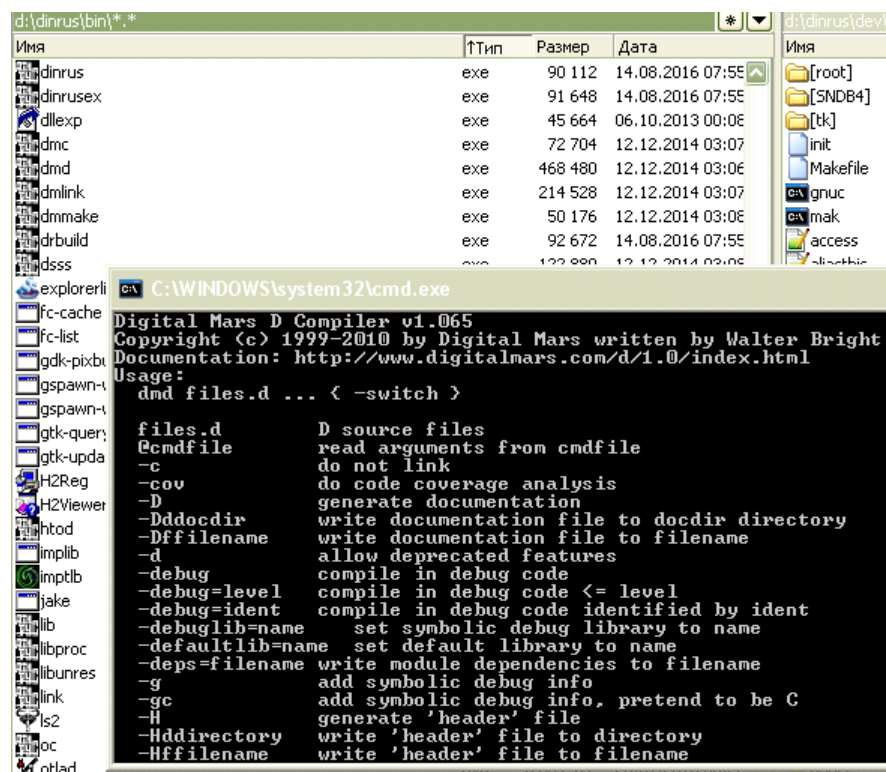
```

## ГЛАВА 1. ВВЕДЕНИЕ В КОДИРОВАНИЕ

### Описание рабочего инструментария

#### 1. Компилятор DMD v1.065

(именно такой используется мною уже несколько лет) был создан Уолтером Брайтом, как инновационный способ упростить судьбу программиста C++.



У него имеется следующий набор функций:

Digital Mars D Compiler v1.065

Copyright (c) 1999-2010 by Digital Mars written by Walter Bright

Documentation: <http://www.digitalmars.com/d/1.0/index.html>

Использование:

```
dmd files.d ... {-переключатель}
```

**files.d** исходные файлы D

**@комфайл** читать аргументы из командного файла

**-c** не производить компоновку

- cov проводить анализ кода (code coverage analysis)
- D генерировать документацию
- Dдокдир записать файл документации в папку докдир
- Dфимяф записать файл документации под названием имяф
- d разрешить компиляцию депрекированного содержимого
- debug компилировать в отладочный код
- debug=уровень компилировать в отладочный код <= уровень
- debug=идент компилировать в отладочный код, идентифицируемый с помощью идент
- debuglib=имя установить указанную библиотеку с отладочными символами
- defaultlib=имя установить библиотеку, используемую по умолчанию
- deps=имяф записать зависимости модуля в файл имяф
- g добавить символьную отладочную информацию
- gs добавить символьную отладочную инфу, в стиле языка C
- H генерировать 'header' (заголовочный) файл
- Hдпапка записать 'header' файл в эту папку
- Hфимяф записать 'header' файл под этим именем
- help вывести справку
- Iпуть где искать импорты
- ignore игнорировать неподдерживаемые прагмы
- inline делать инлайнинг функций
- Jпуть где искать строковые импорты
- Lфлагкомп передать компоновщику данный флаг
- lib генерировать не объектный, а библиотечный файл
- man открыть веб-браузер на странице руководства
- map генерировать файл компоновщика .map
- nofloat не выдавать сноску на плавающую запятую
- O оптимизировать
- o не записывать файл объекта
- odobнап записать файлы объекта/библиотеки в эту папку
- офимяф именовать выходящий файл в имяф
- ор не удалять пути из исходного файла
- profile профилировать производительность времени выполнения генерируемого кода
- quiet подавлять необязательные сообщения
- release компилировать релизную версию
- run исхфл арг... Запуск итоговой программы с передачей аргументов
- unittest компилировать код в юнит-тестах
- v подробно
- v1 язык D версии 1
- version=уровень компилировать код в версии >=уровень
- version=идент компилировать код версии, помеченной идентификатором
- w активировать предупреждения
- wi активировать информационные сообщения
- X генерировать файл JSON
- Xфимяф записать файл JSON в имяф

История умалчивает, почему произошло перемещение на новую, вторую версию.

По моим соображениям, это случилось не просто так... Во второй версии появились новые ключевые слова. Нечто новое было добавлено и в сам компилятор. Факт в том, что масса библиотек оказалась не адаптированной к новому компилятору и подверглась перекомпоновкам.

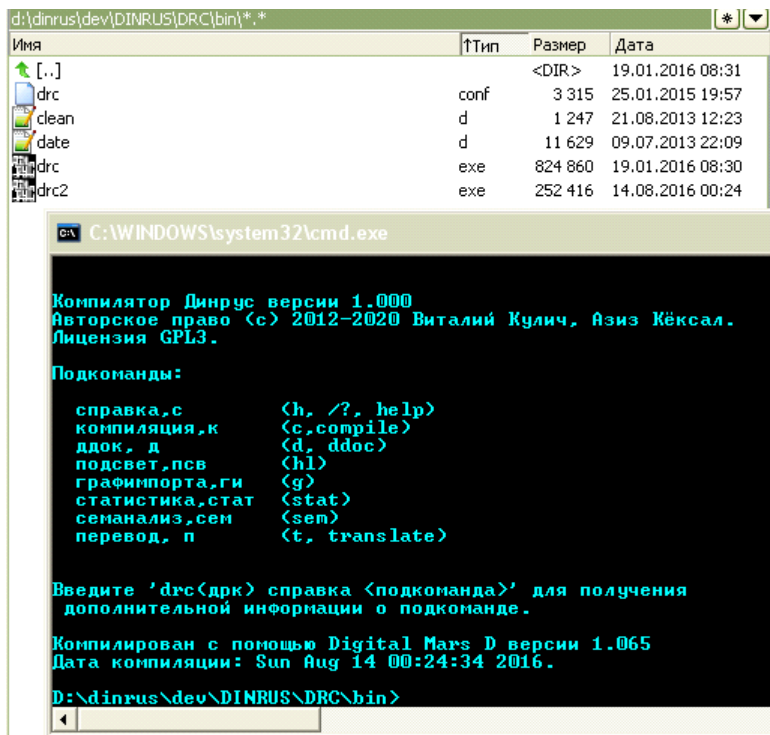
Между тем, некоторые проблемы наблюдаются и сейчас. В принципе, решить их можно, если переписать и дополнить компилятор. Но так, чтобы старое и новое не конфликтовало)))

Мною обнаружен и руссифицирован компилятор DIL, Написанный Азизом Кёксалом на самом D v1.

Однако, после изрядной переработки, я решил присвоить ему другое имя, так как у него ещё не всё есть, чтобы полноценно работать. Имя ему DRC (от "D-in-Russian's Compiler")....

Работу над этим проектом ещё предстоит доделать!))) Вот как он выглядит сейчас:



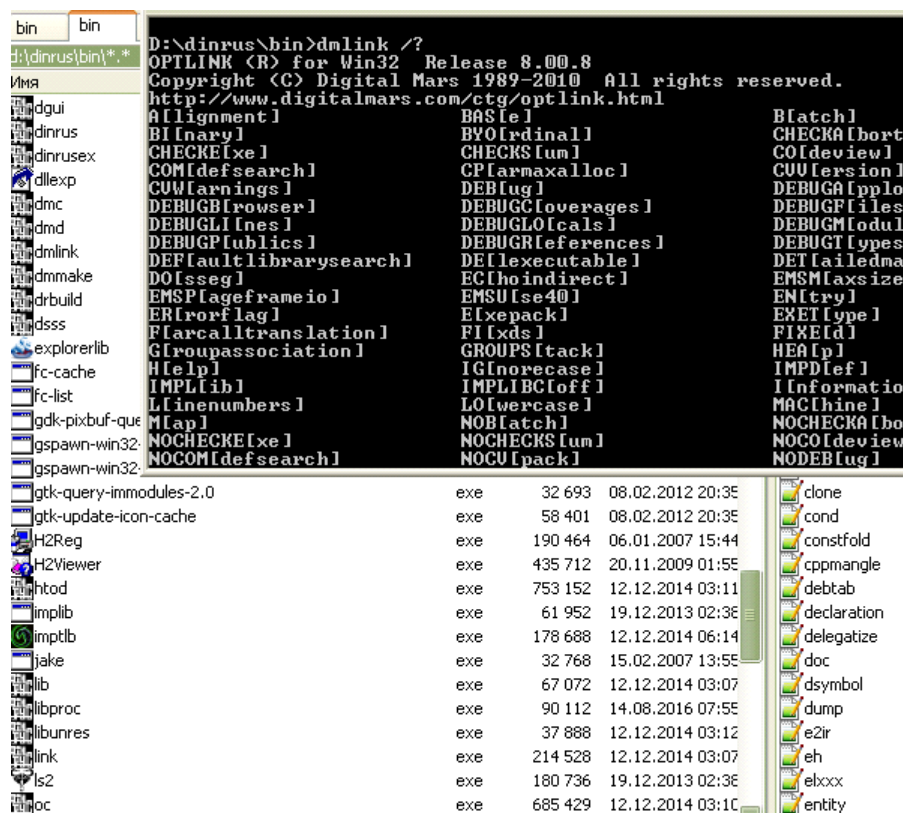


Доработке DRC препятствуют некоторые проблемы в рантайме Динруса, неготовность библиотеки Танго и масса других обстоятельств.

Например, названия модулей принимаются DMD только как `char_t`, в то время как для русского или иного другого языка он должен бы делать это как с `wchar_t`. Именно по этой причине сохраняется `module dinrus`; вместо того, чтобы хотя бы было `module динрус`; !(((

## 2. Компоновщик dmlink

Что касается компоновщика, то предпочтительно было переименовать его в `dmlink`, так как происходит постоянная накладка с другими компоновщиками, указанными в переменной `PATH`. А многие из них называются именно `link`. (((

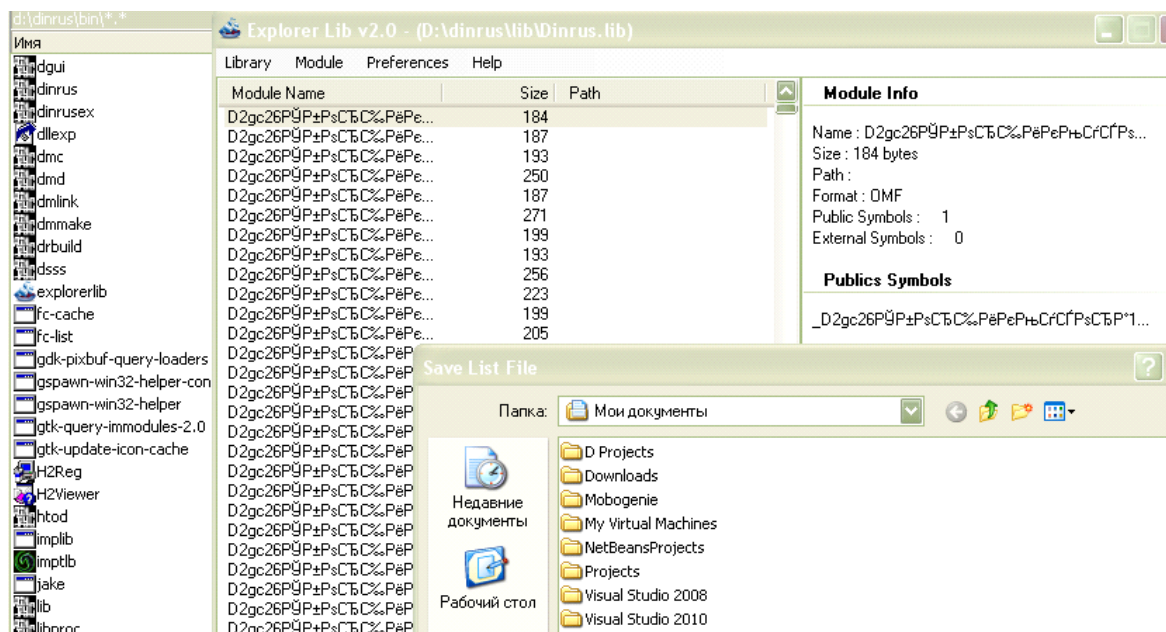


OPTLINK (R) for Win32 Release 8.00.8  
Copyright (C) Digital Mars 1989-2010 All rights reserved.  
<http://www.digitalmars.com/ctg/optlink.html>

A[alignment]	BAS[e]	B[atch]
BI[nary]	BYO[rdinal]	CHECKA[bort]
CHECKE[xe]	CHECKS[um]	CO[deviue]
COM[defsearch]	CP[armaxalloc]	CVV[ersion]
CVW[arnings]	DEB[ug]	DEBUGA[pploader]
DEBUGB[rowser]	DEBUGC[overages]	DEBUGF[iles]
DEBUGLI[nes]	DEBUGLO[cals]	DEBUGM[odules]
DEBUGP[ublics]	DEBUGR[eferences]	DEBUGT[ypes]
DEF[aullibrarysearch]	DE[lexecutable]	DET[ailedmap]
DO[sseg]	EC[hoindirect]	EMSM[axsize]
EMSP[ageframeio]	EMSU[se40]	EN[try]
ER[rerflag]	E[xepack]	EXET[ype]
F[arcalltranslation]	FI[xds]	FIXE[d]
G[roupassociation]	GROUPS[tack]	HEA[p]
H[elp]	IG[norecase]	IMPD[ef]
IMPL[ib]	IMPLIBC[off]	I[nformation]
L[inenumbers]	LO[wercase]	MAC[hine]
M[ap]	NOB[atch]	NOCHECKA[bort]
NOCHECKE[xe]	NOCHECKS[um]	NOCO[deviue]
NOCOM[defsearch]	NOCV[pack]	NODEB[ug]
NODEBUGA[pploader]	NODEBUGB[rowser]	NODEBUGC[overages]
NODEBUGLI[nes]	NODEBUGLO[cals]	NODEBUGP[ublics]
NODEBUGR[eferences]	NODEBUGT[ypes]	NOD[efaultlibrarysearch]
NODEL[executable]	NODET[ailedmap]	NODO[sseg]
NOEC[hoindirect]	NOEMSP[ageframeio]	NOEMSU[se40]
NOER[rerflag]	NOEXE[pack]	NOE[xtdictionary]
NOF[arcalltranslation]	NOFI[xds]	NOG[roupassociation]
NOGROUPS[tack]	NOI[gnorecase]	NOLI[nenumbers]
NOL[ogo]	NOM[ap]	NONA[mes]
NONT[host]	NON[ullsdsseg]	NOP[ackcode]
NOPACKD[ata]	NOPACKF[unctions]	NOPACKI[fnosegments]
NOPAU[se]	NOPR[ompt]	NOR[elocationcheck]
NOREO[rdersegments]	NOSCANLIB	NOSCANLINK
NOWARND[ups]	NOWI[npack]	NOX[ref]
NT[host]	NU[lldsdsseg]	ON[error]
OPT	PAC[kcode]	PACKD[ata]
PACKF[unctions]	PACKI[fnosegments]	PACKS[ize]
PADC[ode]	PADD[ata]	PAG[esize]
PAU[se]	PM[type]	PR[ompt]
RC	RELOC[ationcheck]	REO[rdersegments]
SCANLIB	SCANLINK	SE[gments]
SEGP[ack]	SI[lent]	ST[ack]
STU[b]	SU[bssystem]	T[iny]
U[ppercase]	VERS[ion]	W[arnfixup]
WARND[ups]	WI[npack]	XM[smaxsize]
X[ref]	XN[oignorecase]	XU[ppercase]

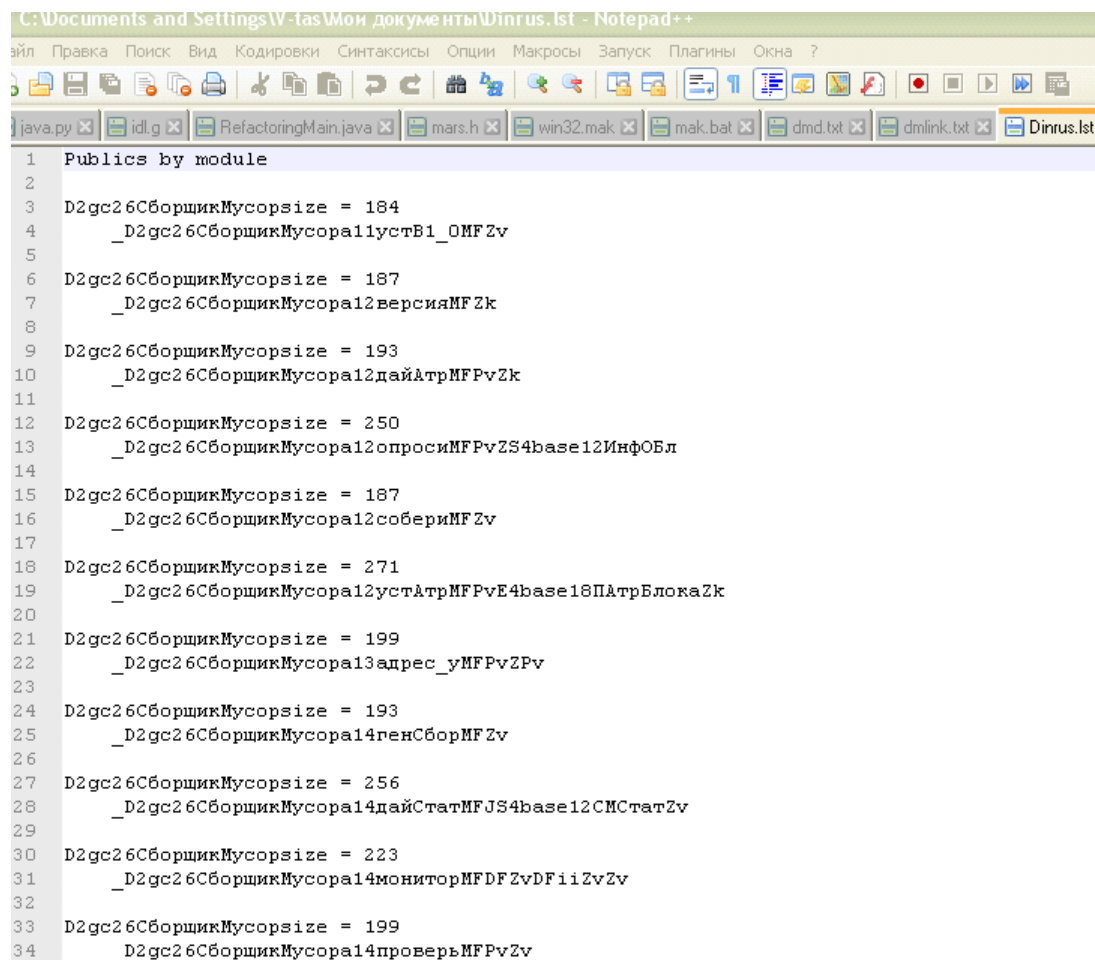
Заглянуть внутрь статической библиотеки позволяет другой, весьма ценный, инструмент, автор которого находится где-то в испаноязычной части мира. К сожалению, исходного кода нет(((

### 3. Утилита Explorer Lib



Остаётся снова возмущаться на судьбу, поскольку большинство инструментов из "наследия" не работает с кодировкой UTF-8, в которой нужно строго сохранять все файлы .d, что позволяет установка свойств в приложении Notepad++.

Именно эта программа позволяет далее успешно заглянуть в листинг и увидеть текст правильно. Хотя "каракатицы" порой выглядят пугающе...



Здесь мы видим "декорированные" символы, способ украшения которых в Ди(нрусе) отличается от C++. Для этого используется программа Demangler, которая правильно расшифровывает записи в статических хранилищах кода....

Это ещё не всё.

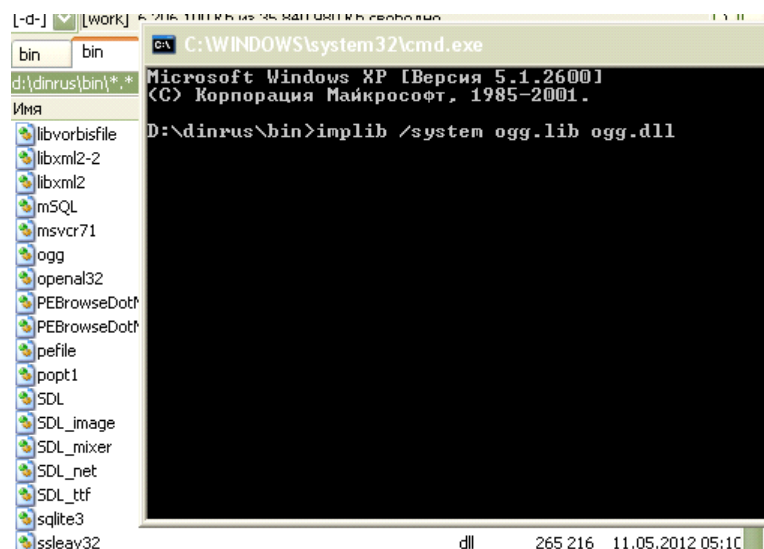
.... Как видно на иллюстрации, библиотека Динрус составлена в формате OMF, что вызывает потребность в иных средствах для преобразований между другими форматами. Такой же формат используют NASM и Delphi, на данный момент известную как Embarcadero RAD Studio 2016. Microsoft CL и GNU компилятор GCC

создают файлы объектов в другом формате, COFF. Этот нюанс также является не очень приятным. Но в наборе папки bin имеется достаточно средств, чтобы преодолевать данные преграды.

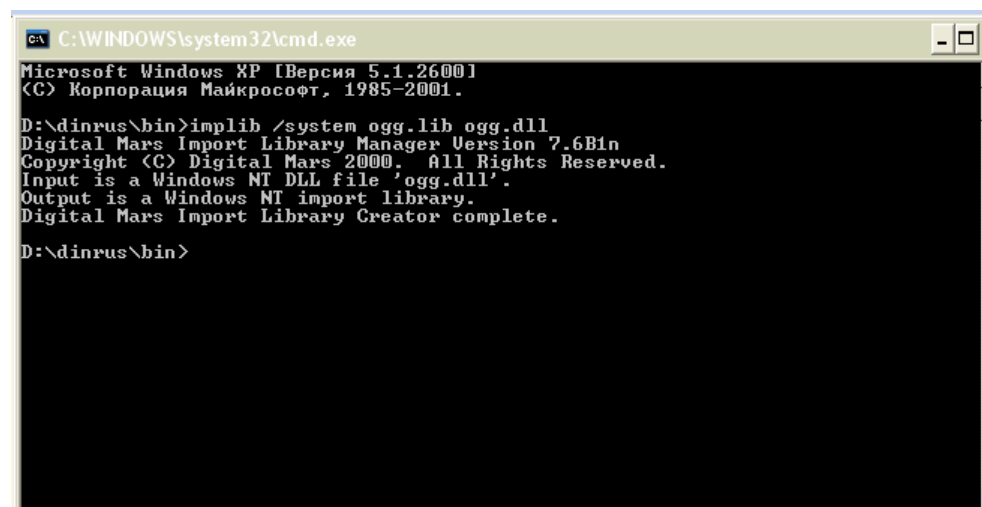
## Описание основных рабочих операций

### 1. Получение библиотеки импорта для той или иной DLL

Строка команды выглядит так: **implib /system В\_КАКУЮ.lib ИЗ\_КАКОЙ.dll**



Выполнение этой операции завершается следующим финалом:



Продвинутый разработчик Нир Зофер придумал другое полезное средство для просмотра динамических библиотек.

DLL Export Viewer						
Файл Правка Вид Настройки Справка						
Имя функции	Адрес	Относительн...	Ординал	Имя фа...	Полный путь	Тип
ogg_packet_clear	0x100028f0	0x000028f0	1 (0x1)	ogg.dll	D:\dinrus\bin\ogg.dll	Экспортируемая функция
ogg_page_bos	0x10001a00	0x00001a00	2 (0x2)	ogg.dll	D:\dinrus\bin\ogg.dll	Экспортируемая функция
ogg_page_checksum_set	0x10001c70	0x00001c70	3 (0x3)	ogg.dll	D:\dinrus\bin\ogg.dll	Экспортируемая функция
ogg_page_continued	0x100019f0	0x000019f0	4 (0x4)	ogg.dll	D:\dinrus\bin\ogg.dll	Экспортируемая функция
ogg_page_eos	0x10001a10	0x00001a10	5 (0x5)	ogg.dll	D:\dinrus\bin\ogg.dll	Экспортируемая функция
ogg_page_granulepos	0x10001a20	0x00001a20	6 (0x6)	ogg.dll	D:\dinrus\bin\ogg.dll	Экспортируемая функция
ogg_page_packets	0x10001b60	0x00001b60	7 (0x7)	ogg.dll	D:\dinrus\bin\ogg.dll	Экспортируемая функция
ogg_page_pageno	0x10001b30	0x00001b30	8 (0x8)	ogg.dll	D:\dinrus\bin\ogg.dll	Экспортируемая функция
ogg_page_serialno	0x10001b00	0x00001b00	9 (0x9)	ogg.dll	D:\dinrus\bin\ogg.dll	Экспортируемая функция
ogg_page_version	0x100019e0	0x000019e0	10 (0xa)	ogg.dll	D:\dinrus\bin\ogg.dll	Экспортируемая функция
ogg_stream_clear	0x10001c00	0x00001c00	11 (0xb)	ogg.dll	D:\dinrus\bin\ogg.dll	Экспортируемая функция
ogg_stream_destroy	0x10001c50	0x00001c50	12 (0xc)	ogg.dll	D:\dinrus\bin\ogg.dll	Экспортируемая функция
ogg_stream_eos	0x10002150	0x00002150	13 (0xd)	ogg.dll	D:\dinrus\bin\ogg.dll	Экспортируемая функция
ogg_stream_flush	0x10001f00	0x00001f00	14 (0xe)	ogg.dll	D:\dinrus\bin\ogg.dll	Экспортируемая функция
ogg_stream_init	0x10001b90	0x00001b90	15 (0xf)	ogg.dll	D:\dinrus\bin\ogg.dll	Экспортируемая функция
ogg_stream_packetin	0x10001d20	0x00001d20	16 (0x10)	ogg.dll	D:\dinrus\bin\ogg.dll	Экспортируемая функция
ogg_stream_packetout	0x10002780	0x00002780	17 (0x11)	ogg.dll	D:\dinrus\bin\ogg.dll	Экспортируемая функция
ogg_stream_packetpeek	0x100028d0	0x000028d0	18 (0x12)	ogg.dll	D:\dinrus\bin\ogg.dll	Экспортируемая функция
ogg_stream_pagein	0x10002400	0x00002400	19 (0x13)	ogg.dll	D:\dinrus\bin\ogg.dll	Экспортируемая функция
ogg_stream_pageout	0x10002100	0x00002100	20 (0x14)	ogg.dll	D:\dinrus\bin\ogg.dll	Экспортируемая функция
ogg_stream_reset	0x10002710	0x00002710	21 (0x15)	ogg.dll	D:\dinrus\bin\ogg.dll	Экспортируемая функция

Эти средства можно скачать по следующей ссылке:

[http://download.nirsoft.net/nirsoft\\_package\\_1.19.69.zip](http://download.nirsoft.net/nirsoft_package_1.19.69.zip)

## 2. Получение шаблона динамического импорта

Среди прочего в Динрус имеется возможность быстрого создания шаблона для динамического импорта кодовых библиотек, основанная на методах, адаптированных по образцу из пакета Derelict.

Для этого существует программка librgos, которая выполняет функцию полуавтомата!))) После её обработки DLL, генерируется полуготовый документ, после чего можно переименовывать функции любым удобным образом.

```

1691
1692 static this() {
1693     SDL.загружай("SDL.dll", &грузи);
1694     SDL.загружай();
1695 }
1696
1697 static ~this() {
1698     SDL.выгружай();
1699 }
1700
1701 debug(Sdl)
1702 {
1703     void main()
1704     {
1705         auto дисп = sdlУстановиВидеоРежим(640,480,0,SDL_HWSURFACE|SDL_DOUBLEBUF);
1706         auto r = ПрямоугСДЛ(0,190,100,100);
1707         auto c = sdlКартируйКЗСА(дисп.формат,255,100,0,255);
1708         while (r.x < дисп.w-100)
1709         {
1710             sdlЗаполниПрямоуг(дисп, пусто, 0);
1711             sdlЗаполниПрямоуг(дисп, &r, c);
1712             sdlФлип(дисп);
1713             r.x++;
1714         }
1715     }
1716

```

Заодно отметим, что `static this(){} - это статический конструктор модуля, который предварительно подготавливает его к предстоящей работе. В данном случае он загружает динамическую библиотеку. Которую в конце работы выгружает статический деструктор модуля.`

Показан лишь отрывок из модуля lib.sdl, который проверяет работу модуля при его отладке. Вообще-то, `debug(Sdl)` надо заменить на `unittest`.

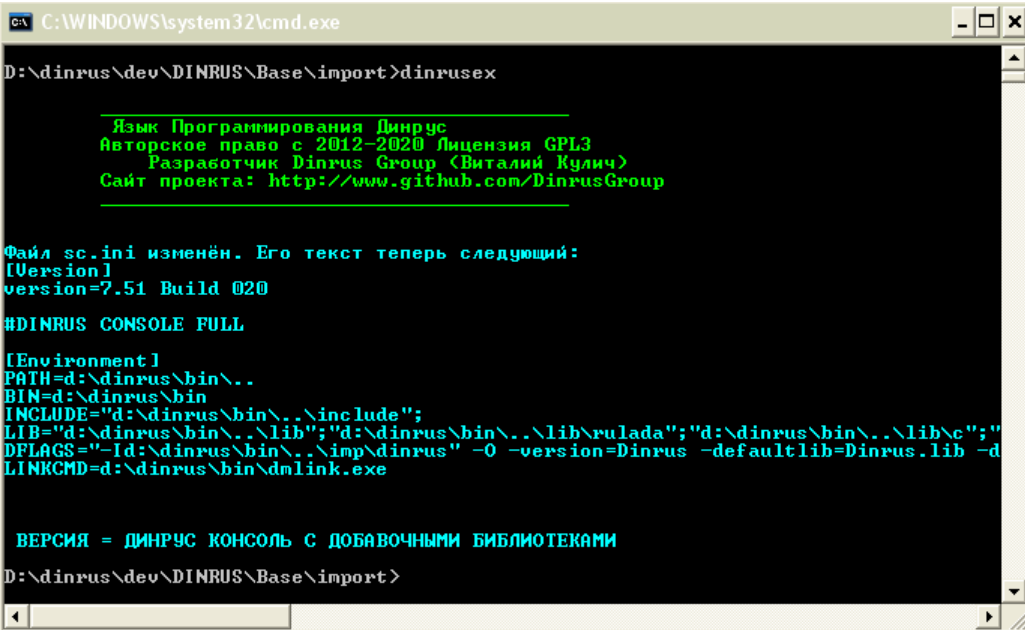
## 2. Использование конфигурационных файлов

В папке бинарных файлов находится файл `sc.ini`, который сообщает компилятору информацию о том, где находятся файлы импорта и библиотеки кодов. Изначально был вопрос с переключением между Phobos и Tango, пока они находились в разных комплектах. Затем появилась необходимость переключаться между версиями D1 и D2.

Итоговое решение оказалось простым. Для каждой из конфигураций написана небольшая программка, которая и выполняет все необходимые манипуляции. На данный момент пока есть 4 рабочих варианта - Рулада, Динрус... Один вариант указывает все библиотеки, которые есть, чтобы при работе избавить от необходимости указывать лишние детали.

Однако в некоторых случаях, например, при компиляции динамической совместной библиотеки, возникает необходимость отключать все, кроме дефолтной, библиотеки. Поэтому и появляется две дополнительные конфигурации.

В bat-файлах достаточно указать rulada, ruladaex, dinrus, dinrusex, чтобы осуществить изменения. Выглядит это в командной строке так:



```
C:\WINDOWS\system32\cmd.exe
D:\dinrus\dev\DINRUS\Base\import>dinrusex

Язык Программирования Динрус
Авторское право с 2012-2020 Лицензия GPL3
Разработчик Dinrus Group <Виталий Кулич>
Сайт проекта: http://www.github.com/DinrusGroup

Файл sc.ini изменён. Его текст теперь следующий:
[Version]
version=7.51 Build 020

#DINRUS CONSOLE FULL

[Environment]
PATH=d:\dinrus\bin\..
BIN=d:\dinrus\bin
INCLUDE="d:\dinrus\bin\..\include";
LIB="d:\dinrus\bin\..\lib";"d:\dinrus\bin\..\lib\rulada";"d:\dinrus\bin\..\lib\c";"
DFLAGS="-Id:\dinrus\bin\..\imp\dinrus" -O -version=Dinrus -defaultlib=Dinrus.lib -d
LINKCMD=d:\dinrus\bin\dmlink.exe

    ВЕРСИЯ = ДИНРУС КОНСОЛЬ С ДОБАВОЧНЫМИ БИБЛИОТЕКАМИ
D:\dinrus\dev\DINRUS\Base\import>
```

Компиляция программ и их замена происходит в подпапке Exe каталога активной разработки dev.

```
module scConfig;
import util.worktools, cidrus, win, stdrus, exception;

version(UCRT)
{
    pragma(lib, "ucrt.lib");
    extern(C)
    {
        int _waccess(wchar* path, int access_mode);
        int _wchmod(wchar* path, int mode);
    }
}
alias пауза пз;
alias раскройГлоб рг;

extern(C):

проц сбросьЦветКонсоли();

const ткст КОНФИГ_РЕБИЛДА;
const ткст КОНФИГ_РУЛАДЫ;
const ткст КОНФИГ_РУЛАДЫДОП;
const ткст КОНФИГ_ДИНРУС;
const ткст КОНФИГ_ДИНРУСДОП;
const ткст КОНФИГ_ДИНРУСДОПГИП;
const ткст КОНФ, СЦ_ИНИ;

static this()
{
    КОНФИГ_РЕБИЛДА =pr(
        "profile=phobos

        compiler=%DINRUS%\dmd.exe
        inifile=%DINRUS%\sc.ini

        exeext=.exe
        objext=obj
        objmodsep=-

        version=DigitalMars
        noversion=GNU
        noversion=linux
```

```

noversion=Unix
noversion=Posix
version=Windows
testversion=Win32
testversion=Win64
version=X86
noversion=PPC
noversion=X86_64
version=D_InlineAsm
version=D_InlineAsm_X86
noversion=D_InlineAsm_PPC
noversion=D_InlineAsm_X86_64
version=LittleEndian
noversion=BigEndian

[compile]
cmd=%DINRUS%\dmd.exe -c $i
response=@

flag=$i
incdir=-I$i
libdir=-L-L$i
optimize=-O
version=-version=$i

[link]
oneatotime=yes
cmd=%DINRUS%\dmd.exe $i -of$o
response=@

libdir=-L+$i\
lib=-L+$i.lib
flag=-L$i
gui=-L/subsystem:windows

[liblink]
safe=yes
oneatotime=yes
cmd=%DINRUS%\lib.exe -c -p512 $o $i
response=@

libdir=
lib=
flag=

[postliblink]
cmd=echo $i

[shliblink]
shlibs=no

[dyliblink]
dylibs=no
");

КОНФИГ_ДИНРУС =pr(
"
[Version]
version=7.51 Build 020

#DINRUS CONSOLE SINGLE

[Environment]
PATH=%DINRUS%\..
BIN=%DINRUS%
INCLUDE="%DINRUS%\..\include\";%INCLUDE%
LIB="%DINRUS%\..\lib\";"%DINRUS%\..\lib\rulada\";"%DINRUS%\..\lib\c\";"%DINRUS%\..\lib\sysimport\"
DFLAGS="-I%DINRUS%\..\imp\dinrus\" -O -version=Dinrus -defaultlib=Dinrus.lib
debuglib=Dinrus.lib
LINKCMD=%DINRUS%\dmlink.exe
");
    КОНФИГ_ДИНРУСДОП = pr(
"
[Version]
version=7.51 Build 020

#DINRUS CONSOLE FULL

[Environment]
PATH=%DINRUS%\..
BIN=%DINRUS%
INCLUDE="%DINRUS%\..\include\";%INCLUDE%
LIB="%DINRUS%\..\lib\";"%DINRUS%\..\lib\rulada\";"%DINRUS%\..\lib\c\";"%DINRUS%\..\lib\sysimport\"

```

```

DFLAGS="-I%DINRUS%\\..\\imp\\dinrus\" -O -version=Dinrus -defaultlib=Dinrus.lib -
debuglib=Dinrus.lib -
L+DinrusWin32.lib+DinrusConc.lib+import.lib+DinrusTango.lib+DinrusDbi.lib+DinrusWinDLL.lib
LINKCMD=%DINRUS%\\dmlink.exe
");
    КОНФИГ_ДИНРУСДОПГИП =pr(
    "

[Version]
version=7.51 Build 020

#DINRUS CONSOLE FULL

[Environment]
PATH=%DINRUS%\\..
BIN=%DINRUS%
INCLUDE=\"%DINRUS%\\..\\include\";%INCLUDE%
LIB=\"%DINRUS%\\..\\lib\";%\"%DINRUS%\\..\\lib\\rulada\";%\"%DINRUS%\\..\\lib\\c\";%\"%DINRUS%\\..\\lib\\sysimport\"
DFLAGS="-I%DINRUS%\\..\\imp\\dinrus\" -O -version=Dinrus -defaultlib=Dinrus.lib -
debuglib=Dinrus.lib -L+DinrusWin32.lib+DinrusConc.lib+import.lib+DinrusTango.lib+DinrusDbi.lib -
L/exet:nt/su:windows:4.0
LINKCMD=%DINRUS%\\dmlink.exe
");
    КОНФИГ_РУЛАДЫ =pr(
    "

[Version]
version=7.51 Build 020

#RULADA CONSOLE SINGLE

[Environment]
PATH=%DINRUS%\\..
BIN=%DINRUS%
INCLUDE=\"%DINRUS%\\..\\include\";%INCLUDE%
LIB=\"%DINRUS%\\..\\lib\";%\"%DINRUS%\\..\\lib\\rulada\";%\"%DINRUS%\\..\\lib\\rulada_eng\";%\"%
DINRUS%\\..\\lib\\c\";%\"%DINRUS%\\..\\lib\\sysimport\"
DFLAGS="-I%DINRUS%\\..\\imp\\rulada_eng\" -O -version=Rulada -defaultlib=rulada.lib -
debuglib=rulada.lib
LINKCMD=%DINRUS%\\dmlink.exe
");
    КОНФИГ_РУЛАДЫДОП =pr(
    "

[Version]
version=7.51 Build 020

#RULADA CONSOLE FULL

[Environment]
PATH=%DINRUS%\\..
BIN=%DINRUS%
INCLUDE=\"%DINRUS%\\..\\include\";%INCLUDE%
LIB=\"%DINRUS%\\..\\lib\";%\"%DINRUS%\\..\\lib\\rulada\";%\"%DINRUS%\\..\\lib\\rulada_eng\";%\"%
DINRUS%\\..\\lib\\c\";%\"%DINRUS%\\..\\lib\\sysimport\"
DFLAGS="-I%DINRUS%\\..\\imp\\rulada_eng\" -O -version=Rulada -defaultlib=rulada.lib -
debuglib=rulada.lib -
L+derelict.lib+tango.lib+auxc.lib+auxd.lib+amigos.lib+arc.lib+gtkD.lib+dgui.lib
LINKCMD=%DINRUS%\\dmlink.exe
");
    КОНФ = pr("%DINRUS%\\..\\etc\\rebuild\\dmd-win");
    ЦЦ_ИНИ = pr("%DINRUS%\\sc.ini");
}

бул проверьЗапись(ткст файл)
{
    version(UCRT) {if( _waccess(вЮ16н(файл), 2) != -1) return да;}
    return нет;
}

    проц версияДинрус()
{
    try
    {
        запиши_в(КОНФ, КОНФИГ_РЕВИЛДА);
        запиши_в(ЦЦ_ИНИ, КОНФИГ_ДИНРУС);
    }
    catch(ФайлИскл фи)
    {
    }
    скажифнс("Файл sc.ini изменён. Его текст теперь следующий: %s", читай_из(ЦЦ_ИНИ));
    нс;
    скажинс(" ВЕРСИЯ = ДИНРУС КОНСОЛЬ ");
}

    проц версияДинрусДоп()
{
    try
    {
        запиши_в(КОНФ, КОНФИГ_РЕВИЛДА);
        запиши_в(ЦЦ_ИНИ, КОНФИГ_ДИНРУСДОП);
    }
    catch(ФайлИскл фи)

```



```

    {
    }
    скажифнс("Файл sc.ini изменён. Его текст теперь следующий: %s", читай_из(СЦ_ИНИ));
    нс;
    скажинс(" ВЕРСИЯ = ДИНРУС КОНСОЛЬ С ДОБАВОЧНЫМИ БИБЛИОТЕКАМИ");
}

проц версияДинрусДоп_ГИП()
{
    try
    {
        запиши_в(КОНФ, КОНФИГ_РЕБИЛДА);
        запиши_в(СЦ_ИНИ, КОНФИГ_ДИНРУСДОПГИП);
    }
    catch(ФайлИскл фи)
    {
    }
    скажифнс("Файл sc.ini изменён. Его текст теперь следующий: %s", читай_из(рг(СЦ_ИНИ)));
    нс;
    скажинс(" ВЕРСИЯ = ДИНРУС ДЛЯ ГИП-ПРИЛОЖЕНИЙ С ДОБАВОЧНЫМИ БИБЛИОТЕКАМИ ");
}

проц версияРулада()
{
    try
    {
        запиши_в(КОНФ, КОНФИГ_РЕБИЛДА);
        запиши_в(СЦ_ИНИ, КОНФИГ_РУЛАДЫ);
    }
    catch(ФайлИскл фи)
    {
    }

    скажифнс("Файл sc.ini изменён. Его текст теперь следующий: %s", читай_из(рг(СЦ_ИНИ)));
    нс;
    скажинс(" ВЕРСИЯ = РУЛАДА КОНСОЛЬ ");
}

проц версияРуладаДоп()
{
    try
    {
        запиши_в(КОНФ, КОНФИГ_РЕБИЛДА);
        запиши_в(СЦ_ИНИ, КОНФИГ_РУЛАДЫДОП);
    }
    catch(ФайлИскл фи)
    {
    }

    скажифнс("Файл sc.ini изменён. Его текст теперь следующий: %s", читай_из(рг(СЦ_ИНИ)));
    нс;
    скажинс(" ВЕРСИЯ = РУЛАДА КОНСОЛЬ С ДОБАВОЧНЫМИ БИБЛИОТЕКАМИ");
}

```

### 3. Средство отладки кода

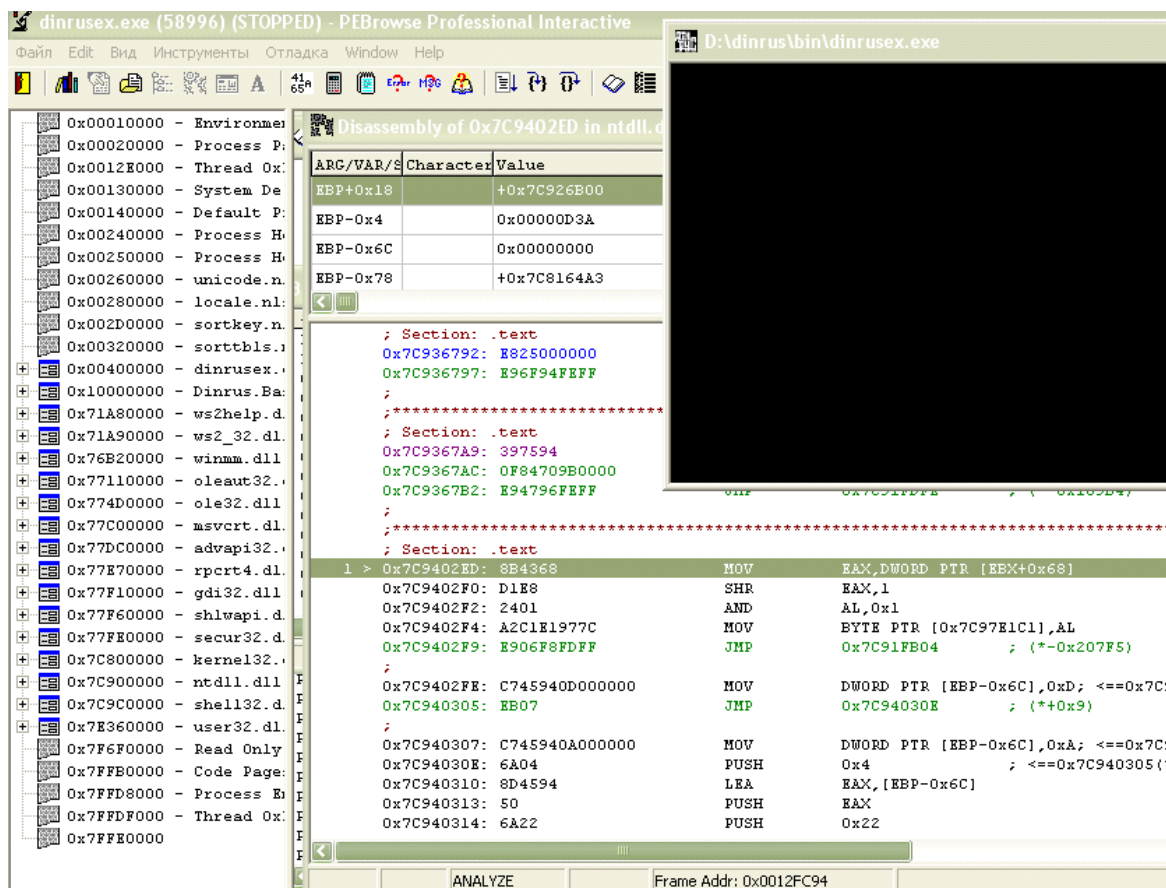
Наибольшее количество времени всегда занимает отладка приложений. А это длительный процесс стабилизации кода, если речь идёт о столь объёмном наборе библиотек, какие есть уже для языка D v1. Более того, каждому программисту приходится постоянно изучать новые и новые коды, так как в общей массе процесс разработки идёт с очень высокой скоростью.

В общем, среди более-менее приемлемых отладчиков, с которыми я сталкивался, следует отметить Ida Pro, Microsoft Visual Studio JIT debugger и PeBrowseDbg.

Для отладки приложения в Динрус нужно применить такую команду:

**otlad имя\_программы**

Появится последний из упомянутых отладчиков, который я частично модифицировал, т.к. этого требовали обстоятельства)))



С этой программой можно долго, весело, пошагово путешествовать по дизассемблированным кодам!!!!)  
Главное, понять их таинственный язык...

## ГЛАВА 2. ВВЕДЕНИЕ В КОДЫ ДИНРУС-РУЛАДЫ (В следующей версии файла)



Если у Вас возникло желание заниматься данной разработкой, появились критические замечания к организации работы или конкретные предложения, пожалуйста, воспользуйтесь средствами связи с Динрус Группой (Dinrus Group):

E-mail:	<a href="mailto:dinruspro@mail.ru">dinruspro@mail.ru</a>
Одноклассники:	<a href="http://ok.ru/vit.klich">http://ok.ru/vit.klich</a>
Телефон:	+7(918)663-79-53, Виталий
Личный сайт:	<a href="http://dinrus.gixx.ru">http://dinrus.gixx.ru</a>
Организация:	<a href="http://github.com/DinrusGroup">http://github.com/DinrusGroup</a>

