Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

D Programming
In nutshell

Jonathan MERCIER

October 17, 2012

fudcon Paris 2012

BY NC SA

Plan

1. Introduction
   - Object
   - Functional
   - Meta-programming
   - Parallelism
   - Ressource Management
   - Contract
   - System and Safe Code
   - Reference and pointer
   - Generics
   - Inference
   - Loops
   - Functions
   - Debugs
   - Versions
   - Requirement
   - Editors
2. Basics
   - My first D program
   - Types
   - Arrays
   - String and characters
   - Const and Immutable
   - Input/Output
   - Algorithm
   - Structure and Class
   - Template
   - Miscellanous
   - Let start it!
3. GTK D
4. Thanks To

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Before starting

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors

Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!

GTK D

Why a new language?

Few significant dates

- C++ 1983

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Why a new language?

Few significant dates

- C++ 1983
- Java 1990

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Why a new language?

Few significant dates

- C++ 1983
- Java 1990
- Python 1995

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Why a new language?

Few significant dates

- C++ 1983
- Java 1990
- Python 1995
- Ruby 1995

D Programming

Jonathan MERCIER

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors

Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!

GTK D

Why a new language?

Few significant dates

- C++ 1983
- Java 1990
- Python 1995
- Ruby 1995
- And now?

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

What is D programming?

D is a modern language programming inspired by:

- C++
- Java
- Haskell
- Python
- Ruby

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Why a new language?

D Combines

- Modeling Power

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors

Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!

GTK D

Why a new language?

D Combines

- Modeling Power
- Modern Convenience

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Why a new language?

D Combines

- Modeling Power
- Modern Convenience
- Native Efficiency

Plan

1. Introduction
   - Object
   - Functional
   - Meta-programming
   - Parallelism
   - Ressource Management
   - Contract
   - System and Safe Code
   - Reference and pointer
   - Generics
   - Inference
   - Loops
   - Functions
   - Debugs
   - Versions
   - Requirement
   - Editors
2. Basics
   - My first D program
   - Types
   - Arrays
   - String and characters
   - Const and Immutable
   - Input/Output
   - Algorithm
   - Structure and Class
   - Template
   - Miscellanous
   - Let start it!
3. GTK D
4. Thanks To

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Object

- Interface

```
interface foo { ...}
```

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Object

- Interface

```
interface foo { …}
```

- class

```
class bar { …}
```

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Object

- Interface

```
interface foo { ...}
```

- class

```
class bar { ...}
```

- inheritance

```
class bar: foo { ...}
```

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Object

- Interface

```d
interface foo { ...}
```

- class

```d
class bar { ...}
```

- inheritance

```d
class bar: foo { ...}
```

- multi class inheritance not allowed, instead used interface.

Plan

1. Introduction
   - Object
   - Functional
   - Meta-programming
   - Parallelism
   - Ressource Management
   - Contract
   - System and Safe Code
   - Reference and pointer
   - Generics
   - Inference
   - Loops
   - Functions
   - Debugs
   - Versions
   - Requirement
   - Editors
2. Basics
   - My first D program
   - Types
   - Arrays
   - String and characters
   - Const and Immutable
   - Input/Output
   - Algorithm
   - Structure and Class
   - Template
   - Miscellanous
   - Let start it!
3. GTK D
4. Thanks To

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Functional

- Data immutability

```d
immutable int[] a = [ 4, 6, 1, 2];
```

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Functional

- Data immutability

```d
immutable int[] a = [ 4, 6, 1, 2];
```

- Pure functions

```d
pure int square(int x) { return x * x; }
```

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Functional

- Data immutability

```
immutable int[] a = [ 4, 6, 1, 2];
```

- Pure functions

```
pure int square(int x) { return x * x; }
```

- Lambda functions

```
a.sort!( (x,y) => x < y ); // [ 1, 2, 4, 6 ]
```

Plan

1. Introduction
   - Object
   - Functional
   - Meta-programming
   - Parallelism
   - Ressource Management
   - Contract
   - System and Safe Code
   - Reference and pointer
   - Generics
   - Inference
   - Loops
   - Functions
   - Debugs
   - Versions
   - Requirement
   - Editors
2. Basics
   - My first D program
   - Types
   - Arrays
   - String and characters
   - Const and Immutable
   - Input/Output
   - Algorithm
   - Structure and Class
   - Template
   - Miscellanous
   - Let start it!
3. GTK D
4. Thanks To

D Programming

Jonathan MERCIER

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors

Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!

GTK D

Meta-programming

Combination of

- templates

D Programming
Jonathan MERCIER
Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Meta-programming

Combination of

- templates
- compile time function execution

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Meta-programming

Combination of

- templates
- compile time function execution
- tuples

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Meta-programming

Combination of

- templates
- compile time function execution
- tuples
- string mixins

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Meta-programming

Code 1: Example

```
template Factorial(ulong n){
  static if(n < 2)
    const Factorial = 1;
  else
    const Factorial = n * Factorial!(n - 1);
}
const ulong var = Factorial!( 8 ); // compute at compile-time
```

Plan

1 Introduction
- Object
- Functional
- Meta-programming
- Parallelism
- Ressource Management
- Contract
- System and Safe Code
- Reference and pointer
- Generics
- Inference
- Loops
- Functions
- Debugs
- Versions
- Requirement
- Editors

2 Basics
- My first D program
- Types
- Arrays
- String and characters
- Const and Immutable
- Input/Output
- Algorithm
- Structure and Class
- Template
- Miscellanous
- Let start it!

3 GTK D

4 Thanks To

D Programming
Jonathan MERCIER
Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Parallelism

- module to use

```d
import std.parallelism;
```

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Parallelism

- module to use

```d
import std.parallelism;
```

- parallel loop

```d
foreach( i; parallel( list ) ){ ...}
```

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Parallelism

- module to use

```
import std.parallelism;
```

- parallel loop

```
foreach( i; parallel( list ) ){ ...}
```

- Pool thread

```
void myfunction( int param1, int param 2 ){ ...}
auto myTask = task!myfunction( param1, param2 );
taskPool.put( myTask );
doSomething();        // another work in parallel
taskPool.finish( true ); // wait alls jobs ending
```

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Plan

1 Introduction
- Object
- Functional
- Meta-programming
- Parallelism
- Ressource Management
- Contract
- System and Safe Code
- Reference and pointer
- Generics
- Inference
- Loops
- Functions
- Debugs
- Versions
- Requirement
- Editors

2 Basics
- My first D program
- Types
- Arrays
- String and characters
- Const and Immutable
- Input/Output
- Algorithm
- Structure and Class
- Template
- Miscellanous
- Let start it!

3 GTK D

4 Thanks To

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Ressource Management

Code 2: Example

```
File f = File( "myfile.txt", "r");
scope(exit) f.close();
lockFile( f );
doFoo( f );
scope(success) doBar( f );
scope(failure) unlock( f );
```

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Plan

1 Introduction
- Object
- Functional
- Meta-programming
- Parallelism
- Ressource Management
- Contract
- System and Safe Code
- Reference and pointer
- Generics
- Inference
- Loops
- Functions
- Debugs
- Versions
- Requirement
- Editors

2 Basics
- My first D program
- Types
- Arrays
- String and characters
- Const and Immutable
- Input/Output
- Algorithm
- Structure and Class
- Template
- Miscellanous
- Let start it!

3 GTK D

4 Thanks To

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Contract

- check a statement

```d
assert( var != null );
```

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Contract

- check a statement

```
assert( var != null );
```

- check before entering into a function

```
double foo ( int a )
in{ assert( a > 0 ); }
body { return a - 2; }
```

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Contract

- check a statement

```
assert( var != null );
```

- check before entering into a function

```
double foo ( int a )
in{ assert( a > 0 ); }
body { return a - 2; }
```

- check at function exit

```
int foo ( int a )
out{ assert( a > 0 ); }
body { return a - 2; }
```

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Contract

- check a statement

```
assert( var != null );
```

- check before entering into a function

```
double foo ( int a )
in{ assert( a > 0 ); }
body { return a - 2; }
```

- check at function exit

```
int foo ( int a )
out{ assert( a > 0 ); }
body { return a - 2; }
```

- -release flag will not compute contract

```
$ ldc2 -release foo.d
```

Plan

1 Introduction
- Object
- Functional
- Meta-programming
- Parallelism
- Ressource Management
- Contract
- System and Safe Code
- Reference and pointer
- Generics
- Inference
- Loops
- Functions
- Debugs
- Versions
- Requirement
- Editors

2 Basics
- My first D program
- Types
- Arrays
- String and characters
- Const and Immutable
- Input/Output
- Algorithm
- Structure and Class
- Template
- Miscellanous
- Let start it!

3 GTK D

4 Thanks To

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

System and Safe Code

System and Safe Code

Safe functions are functions that are statically checked to have no possibility of undefined behavior.

Undefined behavior is often used as a vector for malicious attacks.

Func-tions are marked with at-tributes: @safe, @system, @trusted

Plan

1 Introduction
- Object
- Functional
- Meta-programming
- Parallelism
- Ressource Management
- Contract
- System and Safe Code
- Reference and pointer
- Generics
- Inference
- Loops
- Functions
- Debugs
- Versions
- Requirement
- Editors

2 Basics
- My first D program
- Types
- Arrays
- String and characters
- Const and Immutable
- Input/Output
- Algorithm
- Structure and Class
- Template
- Miscellanous
- Let start it!

3 GTK D

4 Thanks To

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Reference and pointer

- Pointers exist only to create C interface code

```
int* a = cFunction( param );
```

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Reference and pointer

- Pointers exist only to create C interface code

```
int* a = cFunction( param );
```

- ref into function

```
void foo( ref int[] param ) { ...}
```

D Programming

Jonathan MERCIER

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors

Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!

GTK D

Reference and pointer

- Pointers exist only to create C interface code

```
int* a = cFunction( param );
```

- ref into function

```
void foo( ref int[] param ) { ...}
```

- ref into a loop

```
foreach( ref item ; list ){ ...}
```

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Plan

1 Introduction
- Object
- Functional
- Meta-programming
- Parallelism
- Ressource Management
- Contract
- System and Safe Code
- Reference and pointer
- Generics
- Inference
- Loops
- Functions
- Debugs
- Versions
- Requirement
- Editors

2 Basics
- My first D program
- Types
- Arrays
- String and characters
- Const and Immutable
- Input/Output
- Algorithm
- Structure and Class
- Template
- Miscellanous
- Let start it!

3 GTK D

4 Thanks To

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Generic

- class

```
class Foo( T ){ ...}
Foo!int instance = new Foo!(int)( param );
```

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Generic

- class

```
class Foo( T ){ ...}
Foo!int instance = new Foo!(int)( param );
```

- structure

```
struct Foo( T ){ ...}
Foo!int instance = Foo!int( param );
```

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Generic

- class

```
class Foo( T ){ ...}
Foo!int instance = new Foo!(int)( param );
```

- structure

```
struct Foo( T ){ ...}
Foo!int instance = Foo!int( param );
```

- function

```
T foo( T )(T param){ ...}
int var = foo!int( param );
```

D Programming

Jonathan MERCIER

Introduction
- Object
- Functional
- Meta-programming
- Parallelism
- Ressource Management
- Contract
- System and Safe Code
- Reference and pointer
- Generics
- Inference
- Loops
- Functions
- Debugs
- Versions
- Requirement
- Editors

Basics
- My first D program
- Types
- Arrays
- String and characters
- Const and Immutable
- Input/Output
- Algorithm
- Structure and Class
- Template
- Miscellanous
- Let start it!

GTK D

Generic

- class

```d
class Foo( T ){ ...}
Foo!int instance = new Foo!(int)( param );
```

- structure

```d
struct Foo( T ){ ...}
Foo!int instance = Foo!int( param );
```

- function

```d
T foo( T )(T param){ ...}
int var = foo!int( param );
```

- macro

```d
template TFoo( T )( T param ){ immutable T f = param + 3;
        }
int a = TFoo!int( 4 ) ; // return 7 at compile time
```

Plan

1 Introduction
- Object
- Functional
- Meta-programming
- Parallelism
- Ressource Management
- Contract
- System and Safe Code
- Reference and pointer
- Generics
- Inference
- Loops
- Functions
- Debugs
- Versions
- Requirement
- Editors

2 Basics
- My first D program
- Types
- Arrays
- String and characters
- Const and Immutable
- Input/Output
- Algorithm
- Structure and Class
- Template
- Miscellanous
- Let start it!

3 GTK D

4 Thanks To

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Inference

- auto for variable

```
size_t[] list = [ 0, 1, 2, 3, 4];
auto item = list[1]; // item type is size_t
```

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Inference

- auto for variable

```
size_t[] list = [ 0, 1, 2, 3, 4];
auto item = list[1]; // item type is size_t
```

- auto for function

```
auto foo( int param ){ ...}
```

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Plan

1 Introduction
- Object
- Functional
- Meta-programming
- Parallelism
- Ressource Management
- Contract
- System and Safe Code
- Reference and pointer
- Generics
- Inference
- Loops
- Functions
- Debugs
- Versions
- Requirement
- Editors

2 Basics
- My first D program
- Types
- Arrays
- String and characters
- Const and Immutable
- Input/Output
- Algorithm
- Structure and Class
- Template
- Miscellanous
- Let start it!

3 GTK D

4 Thanks To

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Loops

Loops

- for loop

```
for( int i = 0; i < 10; i++ ){ …}
```

- while loop

```
while( isComputing ){ …}
```

- do while

```
do{ …}while( isComputing );
```

- foreach loop

```
foreach( size_t i; list ){ …}
foreach( size_t counter, size_t i = 0; list ){ …}
foreach( counter, i; list ){ …}
```

Plan

1 Introduction
- Object
- Functional
- Meta-programming
- Parallelism
- Ressource Management
- Contract
- System and Safe Code
- Reference and pointer
- Generics
- Inference
- Loops
- Functions
- Debugs
- Versions
- Requirement
- Editors

2 Basics
- My first D program
- Types
- Arrays
- String and characters
- Const and Immutable
- Input/Output
- Algorithm
- Structure and Class
- Template
- Miscellanous
- Let start it!

3 GTK D

4 Thanks To

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Functions

Functions

- classical

```d
void foo( int param ){ ...}
```

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Functions

Functions

- classical

```
void foo( int param ){ ...}
```

- with default values

```
void foo( int param1, int param2 = 3 ){ ...}
```

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Functions

Functions

- classical

```d
void foo( int param ){ ...}
```

- with default values

```d
void foo( int param1, int param2 = 3 ){ ...}
```

- with variadic parameters

```d
void foo( int[] params ... ) {
  foreach( param; params )
    ...// do something
}
```

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Plan

1 Introduction
- Object
- Functional
- Meta-programming
- Parallelism
- Ressource Management
- Contract
- System and Safe Code
- Reference and pointer
- Generics
- Inference
- Loops
- Functions
- Debugs
- Versions
- Requirement
- Editors

2 Basics
- My first D program
- Types
- Arrays
- String and characters
- Const and Immutable
- Input/Output
- Algorithm
- Structure and Class
- Template
- Miscellanous
- Let start it!

3 GTK D

4 Thanks To

D Programming
Jonathan MERCIER
Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Debugs

Debugs

- debug block

```
debug( int param ){ …} // ldc2 −d−debug …
```

- debug line

```
debug writeln( "foo" ) ;
```

- unitest

```
unitest {
  assert( doFoo( x ), true );
  assert( doBar( x ), 3 );
  …
}
```

D Programming
Jonathan MERCIER
Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Debugs

Debugs

- debug block

```
debug( int param ){ …}
```

- debug line

```
debug writeln( "foo" ) ; // ldc2 −d−debug …
```

- unitest

```
unitest {
  assert( doFoo( x ), true );
  assert( doBar( x ), 3 );
  …
}
```

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Debugs

Debugs

- debug block

```
debug( int param ){ …}
```

- debug line

```
debug writeln( "foo" ) ;
```

- unitest

```
unitest { // ldc2 –unittest …
  assert( doFoo( x ), true );
  assert( doBar( x ), 3 );
  …
}
```

Plan

1 Introduction
- Object
- Functional
- Meta-programming
- Parallelism
- Ressource Management
- Contract
- System and Safe Code
- Reference and pointer
- Generics
- Inference
- Loops
- Functions
- Debugs
- Versions
- Requirement
- Editors

2 Basics
- My first D program
- Types
- Arrays
- String and characters
- Const and Immutable
- Input/Output
- Algorithm
- Structure and Class
- Template
- Miscellanous
- Let start it!

3 GTK D

4 Thanks To

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Versions

Versions

- version conditional code block to use

```
version( Windows ){ ...}
else version( linux ){ ...}
else { pragma( msg, "Unknown operating syystem" ); }
```

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Versions

Versions

- own version identifier

```
version( FullApp ){ ...}
else version( DemoApp ){ ...}
else { version = DemoApp ); }
```

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Versions

Versions

- own version identifier

```
version( FullApp ){ …}
else version( DemoApp ){ …}
else { version = DemoApp ); }
```

- from command line give wich version to compile

Code 4: Terminal

```
$ ldc2 -d-version="FullApp" myApp.d
```

Plan

1 Introduction
- Object
- Functional
- Meta-programming
- Parallelism
- Ressource Management
- Contract
- System and Safe Code
- Reference and pointer
- Generics
- Inference
- Loops
- Functions
- Debugs
- Versions
- Requirement
- Editors

2 Basics
- My first D program
- Types
- Arrays
- String and characters
- Const and Immutable
- Input/Output
- Algorithm
- Structure and Class
- Template
- Miscellanous
- Let start it!

3 GTK D

4 Thanks To

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors

Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!

GTK D

Before beginning...

Tools

- Compiler: ldc
- Standard library: phobos
- GUI library: gtkd

Plan

1 Introduction
- Object
- Functional
- Meta-programming
- Parallelism
- Ressource Management
- Contract
- System and Safe Code
- Reference and pointer
- Generics
- Inference
- Loops
- Functions
- Debugs
- Versions
- Requirement
- Editors

2 Basics
- My first D program
- Types
- Arrays
- String and characters
- Const and Immutable
- Input/Output
- Algorithm
- Structure and Class
- Template
- Miscellanous
- Let start it!

3 GTK D

4 Thanks To

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Editors

Tools

- Geany

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Editors

Tools

- Geany
- MonoDevelop with Mono-D

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Editors

Tools

- Geany
- MonoDevelop with Mono-D
- Eclipse with DDT - D Development Tools

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Editors

Tools

- Geany
- MonoDevelop with Mono-D
- Eclipse with DDT - D Development Tools
- Vim + syntastic

D Programming
Jonathan MERCIER
Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Editors

Tools

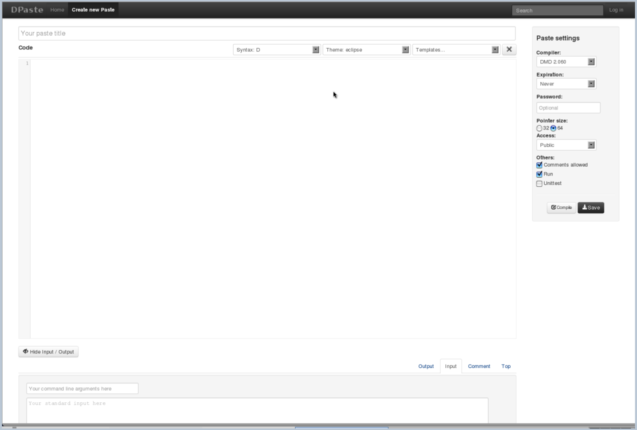
- Geany
- MonoDevelop with Mono-D
- Eclipse with DDT - D Development Tools
- Vim + syntastic
- Emacs + d-mode (GitHub)

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Editors

Online tools: D paste

http://dpaste.dzfl.pl/new


DPaste Home **Create new Paste** Search Log in

Your paste title

Code Syntax: D Theme: eclipse Templates...

Paste settings
Compiler: DMD 2.060
Expiration: Never
Password: Optional
Pointer size: 32 [✓ 64]
Access: Public
Others:
✓ Comments allowed
✓ Run
☐ Unittest

Compile Save

Hide Input / Output

Output Input Comment Top

Your command line arguments here

Your standard input here

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Installation

Code 5: Terminal

```
# yum install ldc-phobos-devel gtkd-devel
```

Plan

1 Introduction
- Object
- Functional
- Meta-programming
- Parallelism
- Ressource Management
- Contract
- System and Safe Code
- Reference and pointer
- Generics
- Inference
- Loops
- Functions
- Debugs
- Versions
- Requirement
- Editors

2 Basics
- My first D program
- Types
- Arrays
- String and characters
- Const and Immutable
- Input/Output
- Algorithm
- Structure and Class
- Template
- Miscellanous
- Let start it!

3 GTK D

4 Thanks To

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

My first D program

Code 6: hello.d

```
module hello;
import std.stdio;

void main () {
  writeln( "Hello world" );
}
```

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

My first D program

Code 7: hello.d

```d
module hello;
import std.stdio;

void main () {
  writeln( "Hello world" );
}
```

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

My first D program

Code 8: hello.d

```d
module hello;
import std.stdio: writeln;

void main () {
  writeln( "Hello world" );
}
```

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

My first D program

Code 9: hello.d

```d
module hello;
import std.stdio: writeln;

void main () {
  writeln( "Hello world" );
}
```

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

My first D program

Code 10: hello.d

```d
module hello;
import std.stdio: writeln;

void main () {
  writeln( "Hello world" );
}
```

D Programming
Jonathan MERCIER
Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Code 11: Terminal

```
$ ldc2 hello.d
$ ./hello
Hello world
```

Plan

1. Introduction
   - Object
   - Functional
   - Meta-programming
   - Parallelism
   - Ressource Management
   - Contract
   - System and Safe Code
   - Reference and pointer
   - Generics
   - Inference
   - Loops
   - Functions
   - Debugs
   - Versions
   - Requirement
   - Editors
2. Basics
   - My first D program
   - Types
   - Arrays
   - String and characters
   - Const and Immutable
   - Input/Output
   - Algorithm
   - Structure and Class
   - Template
   - Miscellanous
   - Let start it!
3. GTK D
4. Thanks To

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Types

Type	bits	Minimum	Maximum
void	Not available	Not available	Not available
byte	8	-128	127
short	16	-32768	32767
int	32	-2147483648	2147483647
long	64	-9223372036854775808	9223372036854775807
ubyte	8	0	255
ushort	16	0	65535
uint	32	0	4294967296
ulong	64	0	18446744073709551615
float	32	$1.18e^{-38}$	$3.40e^{+38}$
double	64	$2.23e^{-308}$	$1.80e^{+308}$
ifloat	32	$1.18e^{-38}$	$3.40e^{+38}$
idouble	64	$2.23e^{-308}$	$1.80e^{+308}$
cfloat	32	$1.18e^{-38}$	$3.40e^{+38}$
cdouble	64	$2.23e^{-308}$	$1.80e^{+308}$
real	128	$3.36e^{-4932}$	$1.19e^{+4932}$
ireal	128	$3.36e^{-4932}$	$1.19e^{+4932}$
creal	28	$3.36e^{-4932}$	$1.19e^{+4932}$
char	utf-8: 8	0	255
wchar	utf-16: 16	0	65535
dchar	utf-32: 32	0	4294967293
bool	8	false	true

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Plan

1 Introduction
- Object
- Functional
- Meta-programming
- Parallelism
- Ressource Management
- Contract
- System and Safe Code
- Reference and pointer
- Generics
- Inference
- Loops
- Functions
- Debugs
- Versions
- Requirement
- Editors

2 Basics
- My first D program
- Types
- Arrays
- String and characters
- Const and Immutable
- Input/Output
- Algorithm
- Structure and Class
- Template
- Miscellanous
- Let start it!

3 GTK D

4 Thanks To

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Static arrays

```
int[3]    a1 = [ 0, 1, 2 ];
int[3][3] a2 = [ [ 0, 1, 2, 3 ], [ 4, 5, 6 ], [ 7, 8, 9 ] ];
a1.length;
```

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Static arrays

```
int[3]    a1 = [ 0, 1, 2 ];
int[3][3] a2 = [ [ 0, 1, 2, 3 ], [ 4, 5, 6 ], [ 7, 8, 9 ] ];
a1.length;
```

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Static arrays

```
int[3]    a1 = [ 0, 1, 2 ];
int[3][3] a2 = [ [ 0, 1, 2, 3 ], [ 4, 5, 6 ], [ 7, 8, 9 ] ];
a1.length; // return array size
```

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Dynamic arrays

```
int[] a1 = [ 0, 1, 2 ];
a1.length;
a1.length = a1.length + 2;
a1.length += 2;
```

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Dynamic arrays

```
int[] a1 = [ 0, 1, 2 ];
a1.length; // return array size
a1.length = a1.length + 2;
a1.length += 2;
```

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Dynamic arrays

```
int[] a1 = [ 0, 1, 2 ];
a1.length;
a1.length = a1.length + 2; // resize array
a1.length += 2;
```

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Dynamic arrays

```d
int[] a1 = [ 0, 1, 2 ];
a1.length;
a1.length = a1.length + 2;
a1.length += 2; // not allowed
```

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Matrix arrays

```
int[][] a1 = new int[][](2,5); // [[0, 0, 0, 0, 0], [0, 0, 0, 0, 0]]
int[][] a2 = [ [ 1, 2, 3 ], [ 4, 5, 6 ], [ 7, 8, 9 ] ];
```

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Arrays

```
int[] a1 = [ 0, 1, 2 ];
a1[0];
a1[0..2];
a1[0..$];
```

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Arrays

```
int[] a1 = [ 0, 1, 2 ];
a1[0]; // return 0
a1[0..2];
a1[0..$];
```

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Arrays

```
int[] a1 = [ 0, 1, 2 ];
a1[0];
a1[0..2]; // return [0, 1]
a1[0..$];
```

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Arrays

```
int[] a1 = [ 0, 1, 2 ];
a1[0];
a1[0..2];
a1[0..$]; // return [0, 1, 2]
```

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Arrays

```
int[] a     = [ 0, 1, 2 ];
int[] b     = a; // 'b' point to 'a' (reference)
int* b_ptr  = b.ptr;
int[] c     = a[ 0 .. 2 ];
int[] d     = a[ 0 .. 2 ].dup;
```

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Arrays

```
int[] a     = [ 0, 1, 2 ];
int[] b     = a;
int* b_ptr = b.ptr; // return pointer to given array
int[] c     = a[ 0 .. 2 ];
int[] d     = a[ 0 .. 2 ].dup;
```

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Arrays

```
int[] a     = [ 0, 1, 2 ];
int[] b     = a;
int*  b_ptr = b.ptr;
int[] c     = a[ 0 .. 2 ]; // is a reference
int[] d     = a[ 0 .. 2 ].dup;
```

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Arrays

```
int[] a     = [ 0, 1, 2 ];
int[] b     = a;
int*  b_ptr = b.ptr;
int[] c     = a[ 0 .. 2 ];
int[] d     = a[ 0 .. 2 ].dup; // is a copy
```

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Vectors

```d
int[] a1 = new int[](2); // [0,0]
a1[] = 1;
a1[]+= 2;
```

D Programming
Jonathan MERCIER
Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Vectors

```
int[] a1 = new int[](2);
a1[] = 1; // [1,1]
a1[]+= 2;
```

D Programming
Jonathan MERCIER
Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Vectors

```d
int[] a1 = new int[](2);
a1[] = 1;
a1[]+= 2; // [3,3]
```

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Associative arrays

```d
string[int] dict;
dict["D"] = 1;
"D" in dict;
```

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Associative arrays

```
string[int] dict;
dict["D"] = 1;
"D" in dict;
```

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Associative arrays

```
string[int] dict;
dict["D"] = 1;
"D" in dict; // true
```

Plan

1 Introduction
- Object
- Functional
- Meta-programming
- Parallelism
- Ressource Management
- Contract
- System and Safe Code
- Reference and pointer
- Generics
- Inference
- Loops
- Functions
- Debugs
- Versions
- Requirement
- Editors

2 Basics
- My first D program
- Types
- Arrays
- String and characters
- Const and Immutable
- Input/Output
- Algorithm
- Structure and Class
- Template
- Miscellanous
- Let start it!

3 GTK D

4 Thanks To

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

String and characters

```
string words = "With double quote, αβγδϵ"; // UTF−8
string words2= words ~ ", and concatenation"; // concat
char  letter= 'a';                            // simple quote
```

Plan

1 Introduction
- Object
- Functional
- Meta-programming
- Parallelism
- Ressource Management
- Contract
- System and Safe Code
- Reference and pointer
- Generics
- Inference
- Loops
- Functions
- Debugs
- Versions
- Requirement
- Editors

2 Basics
- My first D program
- Types
- Arrays
- String and characters
- Const and Immutable
- Input/Output
- Algorithm
- Structure and Class
- Template
- Miscellanous
- Let start it!

3 GTK D

4 Thanks To

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Const and Immutable

Const and Immutable

- im-mutable data: that can-not change.
- const data: can-not be changed
  by the current const ref-er-ence to that data.

Plan

1 Introduction
- Object
- Functional
- Meta-programming
- Parallelism
- Ressource Management
- Contract
- System and Safe Code
- Reference and pointer
- Generics
- Inference
- Loops
- Functions
- Debugs
- Versions
- Requirement
- Editors

2 Basics
- My first D program
- Types
- Arrays
- String and characters
- Const and Immutable
- Input/Output
- Algorithm
- Structure and Class
- Template
- Miscellanous
- Let start it!

3 GTK D

4 Thanks To

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Input/Output

Code 12: Read a file

```
import std.stdio: open, writeln;
...
File f = open( "/path/to/myFile", "r" );
scope(exit) f.close;
foreach( number, line; f )
  writeln( number, line );
...
```

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Input/Output

Code 13: Write a file

```
import std.stdio: open, writeln;
...
File f = open( "/path/to/myFile", "w" );
scope(exit) f.close;
writeln( "something" );
...
```

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors

Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!

GTK D

Input/Output

Code 14: Capture keyboard

```d
import std.stdio: open, writeln;
...
char[] name;
size_t age;
write( "Enter your name : " );
readf( "%s" ~ newline, &name );
write( "How old are you : " );
readf( "%u" ~ newline, &age );
...
```

Plan

1 Introduction
- Object
- Functional
- Meta-programming
- Parallelism
- Ressource Management
- Contract
- System and Safe Code
- Reference and pointer
- Generics
- Inference
- Loops
- Functions
- Debugs
- Versions
- Requirement
- Editors

2 Basics
- My first D program
- Types
- Arrays
- String and characters
- Const and Immutable
- Input/Output
- Algorithm
- Structure and Class
- Template
- Miscellanous
- Let start it!

3 GTK D

4 Thanks To

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Algorithm

Code 15: Searching

```
import std.algorihm: count, countUntil, startsWith, endsWith,
        canFind;
...
int[] list = [ 0, 1, 1, 2, 3, 4, 5, 6, 7, 8, 9 ];
list.count( 1 );
list.countUntil( 2 );
list.startsWith( 0 );
list.endsWith( 9 );
list.canFind( 2 );
```

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Algorithm

Code 16: Searching

```
import std.algorihm: count, countUntil, startsWith, endsWith,
        canFind;
...
int[] list = [ 0, 1, 1, 2, 3, 4, 5, 6, 7, 8, 9 ];
list.count( 1 ); // result => 2
list.countUntil( 2 );
list.startsWith( 0 );
list.endsWith( 9 );
list.canFind( 2 );
```

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Algorithm

Code 17: Searching

```
import std.algorihm: count, countUntil, startsWith, endsWith,
        canFind;
...
int[] list = [ 0, 1, 1, 2, 3, 4, 5, 6, 7, 8, 9 ];
list.count( 1 );
list.countUntil( 2 ); // result => 3
list.startsWith( 0 );
list.endsWith( 9 );
list.canFind( 2 );
```

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Algorithm

Code 18: Searching

```
import std.algorihm: count, countUntil, startsWith, endsWith,
        canFind;
...
int[] list = [ 0, 1, 1, 2, 3, 4, 5, 6, 7, 8, 9 ];
list.count( 1 );
list.countUntil( 2 );
list.startsWith( 0 ); // result => true
list.endsWith( 9 );
list.canFind( 2 );
```

D Programming
Jonathan MERCIER
Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Algorithm

Code 19: Searching

```
import std.algorihm: count, countUntil, startsWith, endsWith,
        canFind;
...
int[] list = [ 0, 1, 1, 2, 3, 4, 5, 6, 7, 8, 9 ];
list.count( 1 );
list.countUntil( 2 );
list.startsWith( 0 );
list.endsWith( 9 ); // result => true
list.canFind( 2 );
```

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Algorithm

Code 20: Searching

```
import std.algorihm: count, countUntil, startsWith, endsWith,
        canFind;
...
int[] list = [ 0, 1, 1, 2, 3, 4, 5, 6, 7, 8, 9 ];
list.count( 1 );
list.countUntil( 2 );
list.startsWith( 0 );
list.endsWith( 9 );
list.canFind( 2 ); // result => true
```

D Programming
Jonathan MERCIER
Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Algorithm

Code 21: Comparison

```
import std.algorihm: min, max;
...
min( 9, 12); // result => 9
max( 9, 12); // result => 12
```

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Algorithm

Code 22: Iteration

```
import std.algorihm: filter, uniq, map, reduce;
int[] list = [ 0, 1, 1, 2, 3, 4, 5, 6, 7, 8, 9 ];
list.filter!( a => a > 6 );
list.uniq( );
list.map!( a => a + 2 );
0.reduce!( (a,b) => a + b )( list );
list.reduce!( min, max )( );
```

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Algorithm

Code 23: Iteration

```
import std.algorihm: filter, uniq, map, reduce;
int[] list = [ 0, 1, 1, 2, 3, 4, 5, 6, 7, 8, 9 ];
list.filter!( a => a > 6 ); // result => [ 7, 8, 9 ]
list.uniq( );
list.map!( a => a + 2 );
0.reduce!( (a,b) => a + b )( list );
list.reduce!( min, max )( );
```

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors

Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!

GTK D

Algorithm

Code 24: Iteration

```
import std.algorihm: filter, uniq, map, reduce;
int[] list = [ 0, 1, 1, 2, 3, 4, 5, 6, 7, 8, 9 ];
list.filter!( a => a > 6 );
list.uniq( ); // [ 0, 1, 2, 3, 4, 5, 6, 7, 8, 9 ]
list.map!( a => a + 2 );
0.reduce!( (a,b) => a + b )( list );
list.reduce!( min, max )( );
```

D Programming
Jonathan MERCIER
Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Algorithm

Code 25: Iteration

```
import std.algorihm: filter, uniq, map, reduce;
int[] list = [ 0, 1, 1, 2, 3, 4, 5, 6, 7, 8, 9 ];
list.filter!( a => a > 6 );
list.uniq( );
list.map!( a => a + 2 ); // [ 2, 3, 3, 4, 5, 6, 7, 8, 9, 10, 11 ]
0.reduce!( (a,b) => a + b )( list );
list.reduce!( min, max )( );
```

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Algorithm

Code 26: Iteration

```
import std.algorihm: filter, uniq, map, reduce;
int[] list = [ 0, 1, 1, 2, 3, 4, 5, 6, 7, 8, 9 ];
list.filter!( a => a > 6 );
list.uniq( );
list.map!( a => a + 2 );
0.reduce!( (a,b) => a + b )( list ); // sum all elements => 49
list.reduce!( min, max )( );
```

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Algorithm

Code 27: Iteration

```
import std.algorihm: filter, uniq, map, reduce;
int[] list = [ 0, 1, 1, 2, 3, 4, 5, 6, 7, 8, 9 ];
list.filter!( a => a > 6 );
list.uniq( );
list.map!( a => a + 2 );
0.reduce!( (a,b) => a + b )( list );
list.reduce!( min, max )( ); // compute in one pass min an max
// min: 0 max: 9
```

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Algorithm

Code 28: Sorting

```
import std.stdio;
import std.algorithm;
import std.array;

void main( ){
    immutable int[] a = [4, 6, 1, 2];
    immutable int[] b = cast(immutable) a.dup
                                          .sort!( (x,y) => x < y )
                                          .array;
    writeln( b ); // [ 1, 2, 4, 6 ]
}
```

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors

Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!

GTK D

Plan

1. Introduction
   - Object
   - Functional
   - Meta-programming
   - Parallelism
   - Ressource Management
   - Contract
   - System and Safe Code
   - Reference and pointer
   - Generics
   - Inference
   - Loops
   - Functions
   - Debugs
   - Versions
   - Requirement
   - Editors
2. Basics
   - My first D program
   - Types
   - Arrays
   - String and characters
   - Const and Immutable
   - Input/Output
   - Algorithm
   - Structure and Class
   - Template
   - Miscellanous
   - Let start it!
3. GTK D
4. Thanks To

D Programming
Jonathan MERCIER
Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Structure

Code 29: Classic implementation

```d
struct MyStruct{
  int    field1;
  float  fiel2;
  string field3;
}
```

D Programming
Jonathan MERCIER
Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Structure

Code 30: With constructor and modifier

```
struct MyStruct{
  private:
  int    __field1;
  float  __fiel2;
  string __field3;

  public:
  this( int f1, float f2, string f3 ){
    __field1 = f1;
    __field2 = f2;
    __field3 = f3;
  }
}
```

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Structure

Code 31: With method and property

```
struct MyStruct{
  private:
  int    _field1;
  float  _field2;
  string _field3;
  public:
  this( int f1, float f2, string f3 ){
    _field1 = f1;
    _field2 = f2;
    _field3 = f3;
  }
  this( MyStruct s ){
    _field1 = s.field1;
    _field2 = s.field2;
    _field3 = s.field3;
  }
  @property int field1( ) const { return _field1; }
  @property void field1( int f1 ){ _field1 = f1; }
  @property float field2( ) const { return _field2; }
  @property void field2( float f2 ){ _field2 = f2; }
  @property string field3( ) const { return _field3; }
  @property void field3( string f3 ){ _field3 = f3; }
  MyStruct dup() const{ return MyStruct( this ); }
}
```

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Class

Code 32: Polymorphism

```
import std.string: format;

Interface IPersonnage{
  string name();
  int health();
  int mana();
}

class BowMan: IPersonnage{
  private:
  string __name;
  int  __health;
  int  __mana;
  public:
  this( string n, int h, int m ){
     __name = n;
     __health = h;
     __mana = m;
  }
  string name() const { return __name; }
  int health() const { return __health; }
  int mana() const { return __mana; }
  override string toString() const{
    return "name: %s point: %d mana: %d".format(__name, __health,
           __mana);
  }
}
```

Plan

1. Introduction
    - Object
    - Functional
    - Meta-programming
    - Parallelism
    - Ressource Management
    - Contract
    - System and Safe Code
    - Reference and pointer
    - Generics
    - Inference
    - Loops
    - Functions
    - Debugs
    - Versions
    - Requirement
    - Editors
2. Basics
    - My first D program
    - Types
    - Arrays
    - String and characters
    - Const and Immutable
    - Input/Output
    - Algorithm
    - Structure and Class
    - Template
    - Miscellanous
    - Let start it!
3. GTK D
4. Thanks To

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Template

Code 33: Function template

```
auto addition(T,U)( T a, U b){
  return a + b;
}
```

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Template

Code 34: Struct template

```
struct TStruct(T) {
  private:
  T _f1;
  public:
  this ( T f1 ){
    _f1 = f1;
  }
}
void main(){
  TStruct!string t1 = TStruct!(string)( "test" );
  auto          t2 = TStruct!int( 5 );
}
```

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Template

Code 35: Class template

```
class TClass(T) {
  private:
  T _f1;
  public:
  this ( T f1 ){
    _f1 = f1;
  }
}
auto c1 = new TClass!int( 2 );
```

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors

Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!

GTK D

Plan

1 Introduction
- Object
- Functional
- Meta-programming
- Parallelism
- Ressource Management
- Contract
- System and Safe Code
- Reference and pointer
- Generics
- Inference
- Loops
- Functions
- Debugs
- Versions
- Requirement
- Editors

2 Basics
- My first D program
- Types
- Arrays
- String and characters
- Const and Immutable
- Input/Output
- Algorithm
- Structure and Class
- Template
- Miscellanous
- Let start it!

3 GTK D

4 Thanks To

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Miscellanous

```
int β= 5; // variable name can use UTF−8 char
int i = 1_000_000; // easy to read number
```

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Plan

1. Introduction
   - Object
   - Functional
   - Meta-programming
   - Parallelism
   - Ressource Management
   - Contract
   - System and Safe Code
   - Reference and pointer
   - Generics
   - Inference
   - Loops
   - Functions
   - Debugs
   - Versions
   - Requirement
   - Editors
2. Basics
   - My first D program
   - Types
   - Arrays
   - String and characters
   - Const and Immutable
   - Input/Output
   - Algorithm
   - Structure and Class
   - Template
   - Miscellanous
   - Let start it!
3. GTK D
4. Thanks To

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

What Are You Waiting For?

- Web site: `dlang.org`

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

What Are You Waiting For?

- Web site: `dlang.org`
- Community: `forum.dlang.org`

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

What Are You Waiting For?

- Web site: `dlang.org`
- Community: `forum.dlang.org`
- Contribute: `www.github.com/D-Programming-Language`

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

What Are You Waiting For?

- Web site: `dlang.org`
- Community: `forum.dlang.org`
- Contribute: `www.github.com/D-Programming-Language`
- irc on freenode #d

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors

Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!

GTK D

What Are You Waiting For?

- Web site: `dlang.org`
- Community: `forum.dlang.org`
- Contribute: `www.github.com/D-Programming-Language`
- irc on freenode #d
- french speaker on jabber d-fr@chat.jabberfr.org

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

GTK D

Code 36: First graphical application

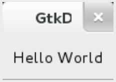
```
module myFirstGUI;

import gtk.MainWindow;
import gtk.Label;
import gtk.Main;

class myFirstGUI: MainWindow{
  this(){
    super("GtkD");
    setBorderWidth(10);
    add(new Label("Hello World"));
    showAll();
  }
}

void main(string[] args){
  Main.init(args);
  new HelloWorld();
  Main.run();

}
```

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

GTK D

Code 37: Terminal

```
$ ldc2 -L-lgtkd -L-ldl myFirstGUI.d
$ ./myFistGUI
```



Hello World

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

GTK D

Code 38: First graphical application

```
module myFirstGUI;

import gtk.MainWindow;
import gtk.Label;
import gtk.Main;

pragma(lib, "gtkd");

class myFirstGUI: MainWindow{
  this(){
    super("GtkD");
    setBorderWidth(10);
    add(new Label("Hello World"));
    showAll();
  }
}

void main(string[] args){
  Main.init(args);
  new myFirstGUI();
  Main.run();

}
```

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

GTK D

Code 39: First graphical application

```
module myFirstGUI;

import gtk.MainWindow;
import gtk.Label;
import gtk.Main;

pragma(lib, "gtkd");

class myFirstGUI: MainWindow{
  this(){
    super("GtkD");
    setBorderWidth(10);
    add(new Label("Hello World"));
    showAll();
  }
}

void main(string[] args){
  Main.init(args);
  new myFirstGUI();
  Main.run();

}
```

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

GTK D

Code 40: First graphical application

```
module myFirstGUI;

import gtk.MainWindow;
import gtk.Label;
import gtk.Main;

pragma(lib, "gtkd");

class myFirstGUI: MainWindow{
  this(){
    super("GtkD");
    setBorderWidth(10);
    add(new Label("Hello World"));
    showAll();
  }
}

void main(string[] args){
  Main.init(args);
  new myFirstGUI();
  Main.run();

}
```

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

First graphical Application

Code 41: Terminal

```
$ ldc2 -L-ldl myFirstGUI.d
$ ./myFistGUI
```

D Programming
Jonathan MERCIER
Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

GTK D

Code 42: First graphical application

```d
module myFirstGUI;

import gtk.MainWindow;
import gtk.Label;
import gtk.Main;

pragma(lib, "gtkd");
version( Linux ) pragma(lib, "dl");

class myFirstGUI: MainWindow{
  this(){
    super("GtkD");
    setBorderWidth(10);
    add(new Label("Hello World"));
    showAll();
  }
}

void main(string[] args){
  Main.init(args);
  new myFirstGUI();
  Main.run();

}
```

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors

Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!

GTK D

First graphical Application

Code 43: Terminal

```
$ ldc2 myFirstGUI.d
$ ./myFistGUI
```

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Contact

Comments, suggestions or bug reports ?

Please send a mail at:
`bioinfornatics@fedoraproject.org`

Introduction
Object
Functional
Meta-programming
Parallelism
Ressource Management
Contract
System and Safe Code
Reference and pointer
Generics
Inference
Loops
Functions
Debugs
Versions
Requirement
Editors
Basics
My first D program
Types
Arrays
String and characters
Const and Immutable
Input/Output
Algorithm
Structure and Class
Template
Miscellanous
Let start it!
GTK D

Thanks to

- To your attention
- French fedora community
- D community
- Mohamed El Morabity