D Idioms

Convenience. Power. Efficiency.

Stefan Rohe

28.01.2016, Munich

Expressiveness

- Python:

```
timedelta(seconds=42)
```

- C++:

```
using namespace boost::posix_time;
seconds(42);
chrono::seconds operator "" s(long secs);
42_s; // user defined literals c++14
```

- D:

```
42.seconds // UFCS
           // uniform function call syntax
f(g(x)) → x.g.f
```

Iterations

- `for (uint i=0; i<10; i++) {}`
  - Every for is a „bug“
- `foreach (i; 0..10) {}`
  - for every range/container/slice/...
  - Even foreach is a „bug“
- `iota(0, 10).each  /  iota(0, 10).map`
  - → Pipeline programming

Pipeline programming

```
import std.algorithm, std.conv, std.functional,
  std.math, std.regex, std.stdio;

alias round = pipe!(to!real, std.math.round, to!string);
static reFloat = ctRegex!`[0-9]+\.[0-9]+`;

void main() {
    // round floating numbers, ignore the rest
    stdin
        .byLine
        .map!(l => l.replaceAll!(c => c.hit.round)
           (reFloat))
        .each!writeln;
}
```

- Draft the pipeline. For production @nogc pipeline.
- No need for python prototyping

Compile Time Execution

```
import pegged.grammar;
mixin(grammar("
  Expr      < Factor AddExpr*
  AddExpr   < ('+' / '-') Factor
  Factor    < Primary MulExpr*
  MulExpr   < ('*' / '/') Primary
  Primary   < Parens / Number / Variable / '-'
  Primary
  Parens    < '(' Expr ')'
  Number    <~ [0-9]+
  Variable  <- Identifier
"));
```

Compile Time Execution

```
// Parsing at compile time
static parseTree1 = Expr.parse(
  "1 + 2 – (3*x-5)*6");
pragma(msg, parseTree1.capture);
// Parsing at run time
auto parseTree2 = Expr.parse(
  readln());
writeln(parseTree2.capture)
```

Destroy!

Links:

http://dlang.org

D Idioms (http://p0nce.github.io/d-idioms/)

C to D (http://dlang.org/ctod.html)

C++ to D (http://dlang.org/cpptod.html)

Alexandrescu: Three Cool Things About D (https://www.youtube.com/watch?v=FdpaBHyQNco)

Contact:

thiSnkP@hoAtmaMil.de (remove SPAM)

Design By Introspection

```
bool reallocate(ref void[] b, size_t newSize)
{
  if (newSize == 0) {
    static if (hasMember!(typeof (this),
      "deallocate"))
      deallocate(b);
    return true;
  }
  if (b is null) {
    b = allocate(newSize);
    return b !is null;
  } ...
```

Other cool Things

- scope(exit)
- alias this
- eponymous templates
- Voldemort Types
- @nogc
- built in unittests
- implicit Conversion of User Types
- contracts
- uda
- pure
- std.variant