TNTNET
Web dynamite

Michal Hrušecký

Writing a web service

PHP

- old and everybody claims to knows it
- syntax similar to many languages but still different

RoR

- used to be the cool kid
- plenty of generators
- easy to start, hard to maintain

Node.js

- new even cooler kid
- javascript all the way

Tntnet

- web development framework
- contains web template system
- contains routing tables
- uses C++ as a programing language
  ⇒ use language you are familiar with
  ⇒ many libraries ready to use
- has to be precompiled
  ⇒ fast
- output can be
  - standalone application
  - shared library for web server

Tntnet templating system

Basically html.

Special tags for various purposes:

- <%pre></%pre> - includes and defines and similar
- <%cpp></%cpp> or <{ code }> - C++ code
- <%args></%args> - query arguments
- <%session></%session> - session variables definitions
- <$ data $> - output the content of the variable/expression
- <? (cond == true) ? data ?> - conditional expression output

Variables

Various scopes:

- component - one component in ecpp file
- page - all components in the same ecpp file
- global - globally available

Various lifetimes:

- request
- session
- thread
- application

Tntnet example - Hello World

```
<# This is a simple hello-world-application #>
<%args>
name; // define query-parameter
</%args>
<html>
 <head>
  <title>Hello World-application for tntnet</title>
 </head><body bgcolor="#FFFFFF">
  <h1>Hello <$ name.empty() ? "World" : name $></h1>
  <form>What's your name?
   <input type="text" name="name" value="<$name$>">
   <br/>
   <input type="submit">
  </form>
 </body>
</html>
```

Tntnet example - Standalone

```
#include <tnt/tntnet.h>

int main(int argc, char* argv[])
{
  try
  {
    tnt::Tntnet app;
    app.listen(8000);
    app.mapUrl("^/", "hello");
    app.run();
  }
  catch (const std::exception& e)
  {
    std::cerr << e.what() << std::endl;
  }
}
```

Tntnet - routing

```
<!-- We are getting authorized -->
<mapping>
  <target>login@test</target>
  <url>^/login$</url>
</mapping>
<!-- Modifying requests have to be authenticated -->
<mapping>
  <target>auth-verify@test</target>
  <method>(POST|PUT|DELETE)</method>
</mapping>
<!-- Make sure everybody speaks json from now on -->
<mapping>
  <target>json@bios_web</target>
</mapping>
```

Tntdb

- database abstraction library
- plugins for SQLite, MySQL, PostgreSQL or Oracle
- one C++ API with cool functions
  - cached prepared statements
  - connection pool
  - conversion from and to basic datatypes
  - possibility to extend conversion to your own classes
  - integrate dtransactions support

Tntdb example - hello world

```
void insData()
{
  std::string url = "mysql:db=mydb;host=192.168.0.1";
  tntdb::Connection conn;
  conn = tntdb::connectCached(url);

  tntdb::Statement st = conn.prepareCached(
    "insert into table values (:v1, :v2)");

  st.setInt("v1", 1)
    .setString("v2", "hi")
    .execute();

  st.setInt("v1", 2)
    .setString("v2", "world")
    .execute();
}
```

Tntdb example - transaction

```
#include <tntdb/transaction.h>

void doSomeModifications(tntdb::Connection conn)
{
  tntdb::Transaction trans(conn);

  // do some modifications in the database here:
  conn.execute(...);
  conn.prepare("...").set("col1", value).execute();

  trans.commit();
}
```

Cxxtools

- library used by Tntnet and Tntdb
  - can be used even without them

- plenty of useful classes
  - threadpool
  - thread abstraction
  - serialization (json, csv, xml, ...)
  - regexp matching
  - iniparser
  - loger
  - ...

What next?

Web page with description:

http://tntnet.org/

What can be in template:

```
man 7 ecpp
```