This slide is left intentionally blank

Tapir

A Thrift-based rapid API framework

What is Thrift?

- An Interface Definition Language (IDL) to define and create services for numerous languages.
- https://thrift.apache.org/
- Developed by Facebook
- Open sourced in April, 2007
- Used by last.fm, reCaptcha, Cassandra

Advantages

- Generates both the server and client interfaces in a consistent manner
- Type safety: rudimentary type/ structure checking
- Typed exceptions
- Built in versioning
- Mature, well used and tested software
- Supports multiple transports and protocols
- Generated code in C++, Java, Python, PHP, Ruby, Erlang, Perl, Haskell, C#, Cocoa, JavaScript, Node.js and other languages

Disadvantages

- Poorly documented specification
  The original Thrift White Paper from 2007 is the best place for information, and lacks clear direction for the framework
- Yet another dependency for creating a service
- Clients and servers must know the schema to parse the protocol

Types

…your standard base types…

- Bool
  bool
- Numbers:
  - Signed integer (8, 16, 32 and 64 bit)
    byte, **i16**, **i32**, **i64**
  - Floating point double (64 bit)
    double
- Strings (utf8 or binary)
  string, **binary**

…your standard containers…

- List
  An array of the same type of object
  list<string>

- Set
  Like a list, but items are unique
  set<i32>

- Map
  Like a Perl hash or JSON object, maps are a series of key/value pairs. Unlike other languages, the key may be any type.
  map<string, i32>: “Map a string to a number”
  map<list<i32>,string>: “Map a list of numbers to a string”

...some definitions...

- Enum
  enum site { shutterstock, bigstock, opal }
- Typedef
  typedef string username

... and my favorite, the struct

- Comprised of fields. Each field has:
- Mostly required field identifier (1, 2, 3, ...)
- Optional or required
- A type
- A unique name
- Optionally a default value

```
struct account {
    1: string  user,
    2: string  pass,
    3: optional bool is_admin
}
```

Typedefs make the schema more readable and extendable

```
typedef string username
typedef string password

struct account {
    1: username   user,
    2: password   pass,
    3: optional bool is_admin
}
```

Service

- Services are comprised of methods
- They may extend the behavior of another service

Method

- Have a typed return value or void
- Argument list is a struct
- Optional throw list is a struct

```
service Accounts {
    account createAccount {
        1: username   user,
        2: password   pass,
        3: optional bool is_admin
    } throws {
        1: InvalidUsername user
    }
}
```

So what is Tapir?

It’s a way to embed business logic in the Thrift document

```
/*
    Your Shutterstock username
    @validate length 1-32
    @validate regex /^[^ ]$/
*/
typedef string username

// @validate range 100-60000
typedef i32 account_id
```

It’s a rapid service builder, handling the transport so you can focus on code

```
package MyAPI;
use Moose;
use Tapir::Server::Handler::Signatures;
extends 'Tapir::Server::Handler::Class';

method createAccount ($username, $password) {
    # ... add your logic here ...
    $call->set_result($return_value);
}

1;
```

It’s a way to connect the service methods to RESTful routes via Dancer

```
/*
    Create a new Shutterstock account
    @rest POST /accounts/:site
*/
account createAccount {
    1: string site,
    2: username  user,
    3: password  pass
}
```

```
package MyApp;
use Dancer ':syntax';
use Dancer::Plugin::Tapir;

setup_tapir_handler
    thrift_idl          => ‘thrift/service.thrift’,
    handler_class   => ‘MyApp::Handler’;

true;
```

It provides easy to use documentation

Demo Time!

Tapir Todo List

- Javascript and Perl client code
- Browser API tool, integrated into the Dancer plugin (partly done)
- Better exception handling
- Make Dancer plugin output context-sensitive data (based on Accept: header)
- Express testable payloads via the Thrift file for NTF and unit tests
- Easy version control

```
account createAccount {
    1: username   user,
    2: optional bool is_admin,   // @version 1
    3: list<role> roles,         // @version 2
}

method createAccount : version 1 ($is_admin) { ... }

method createAccount : version 2 ($roles) { ... }
```

For more information...

- Checkout the following modules on Github and CPAN:
  - Thrift::Parser
  - Tapir
  - Dancer::Plugin::Tapir