

Web.pm

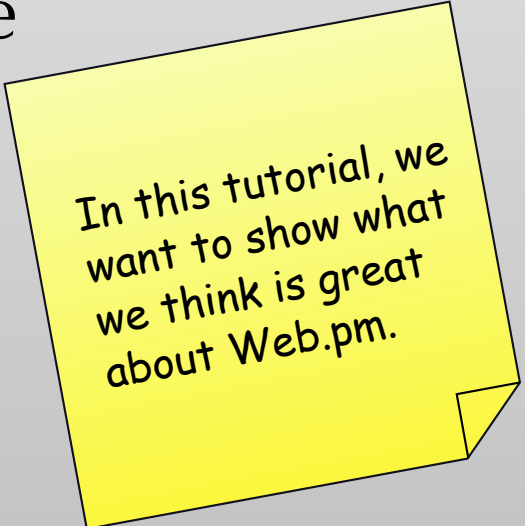


is

neat

neat, adjective, -er, -est

1. pleasingly orderly
2. of a simple design
3. cleverly effective
4. great; wonderful; fine



In this tutorial, we
want to show what
we think is great
about Web.pm.

Web.pm CORE lets you construct web applications out of anything

```
class HelloWorld {  
  method postcircumfix:<( )>($env) {  
    [200,  
     {'Content-Type' => 'text/plain'},  
     ['Hello world!']  
    ]  
  }  
}
```

Any callable object can be used as a web application by Web.pm. It should return status, header, and body.

Web.pm CORE

separates

the application

from

the server engine.



It's easy to switch
to another server
engine, and back-
end code won't
get tangled with
application code.

More help from Web::Request and Web::Response



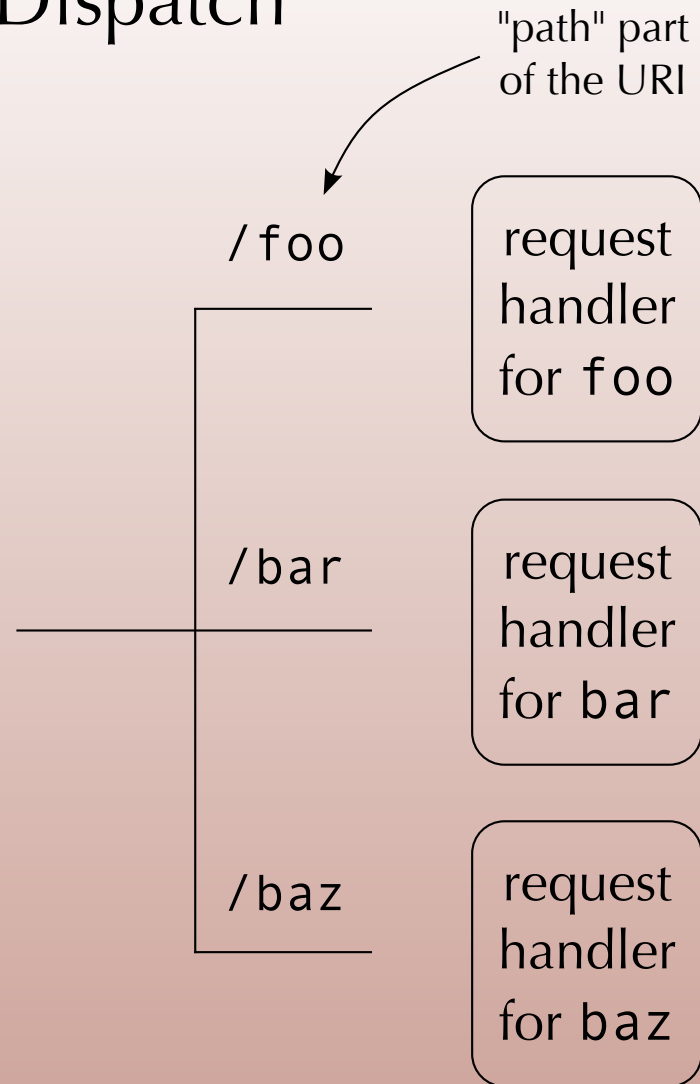
`Web::Request` helps make sense of the `%env` hash, and provides convenience methods to probe it.

`Web::Response` helps prepare the output from the web app. It also has utility methods for various things.

This design is ~~stolen~~ admirably borrowed from Ruby's Rack, by the way. As is the rest of Web.pm CORE.

Dispatch

request
comes in



Dispatching on the URI of the request is a common thing to do in a web app, simply because web apps tend to do more than one thing.

Easy dispatch with Astaire

```
use v6;  
use Astaire;  
  
get '/hi' => {  
    "OH HAI"  
}
```

"Wait, where'd
Web::Request and
Web::Response go?"
you might ask. Well,
Astaire uses them
internally.