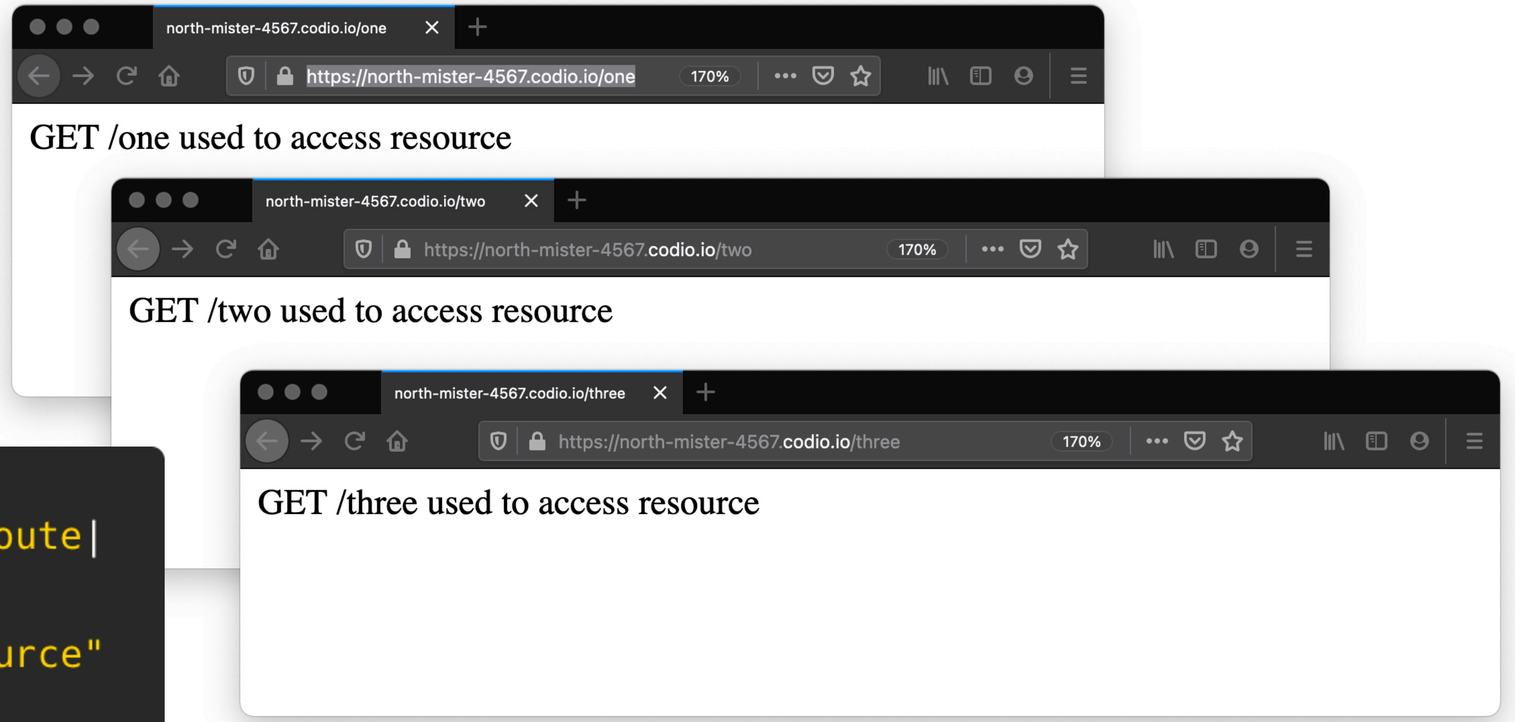
COM 1001

Introduction to Software Engineering

Professor Phil McMinn

Advanced Routes

Multiple Routes to a Single Block




```ruby
["/one", "/two", "/three"].each do |route|
  get route do
    "GET #{route} used to access resource"
  end
end
```

routes/multi_route_resource.rb

Routes with Wildcards

```ruby
get "/date/*-*-*" do
  y = params[:splat][0]
  m = params[:splat][1]
  d = params[:splat][2]

  "The date requested is #{y}-#{m}-#{d}"
end
```

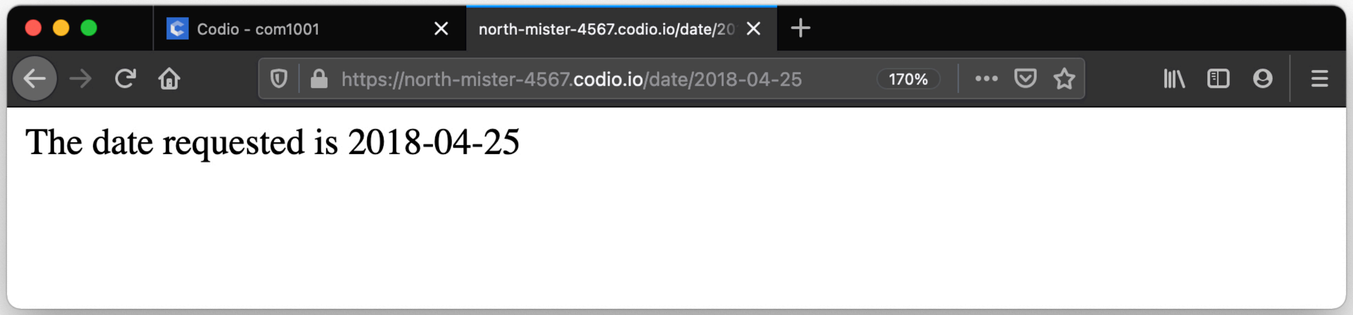
routes/wildcards.rb

Routes can contain wildcards using the splat (*) character. The splat character will match any character up until the next one specified (in this case, the – character)

The characters matching each splat may obtained via the `params` hash, using the `:splat` key, as shown here.

Note that we haven't **validated** these inputs. In this example – we could type any junk between the -s and it would be accepted.

This is the response for the URL `date/2018-04-25`:



The First Match Wins

```ruby
get "/*" do
  "I am the god of all routes. Nothing shall get past me"
end

get "/introverted-route" do
  "You will never see me"
end
```

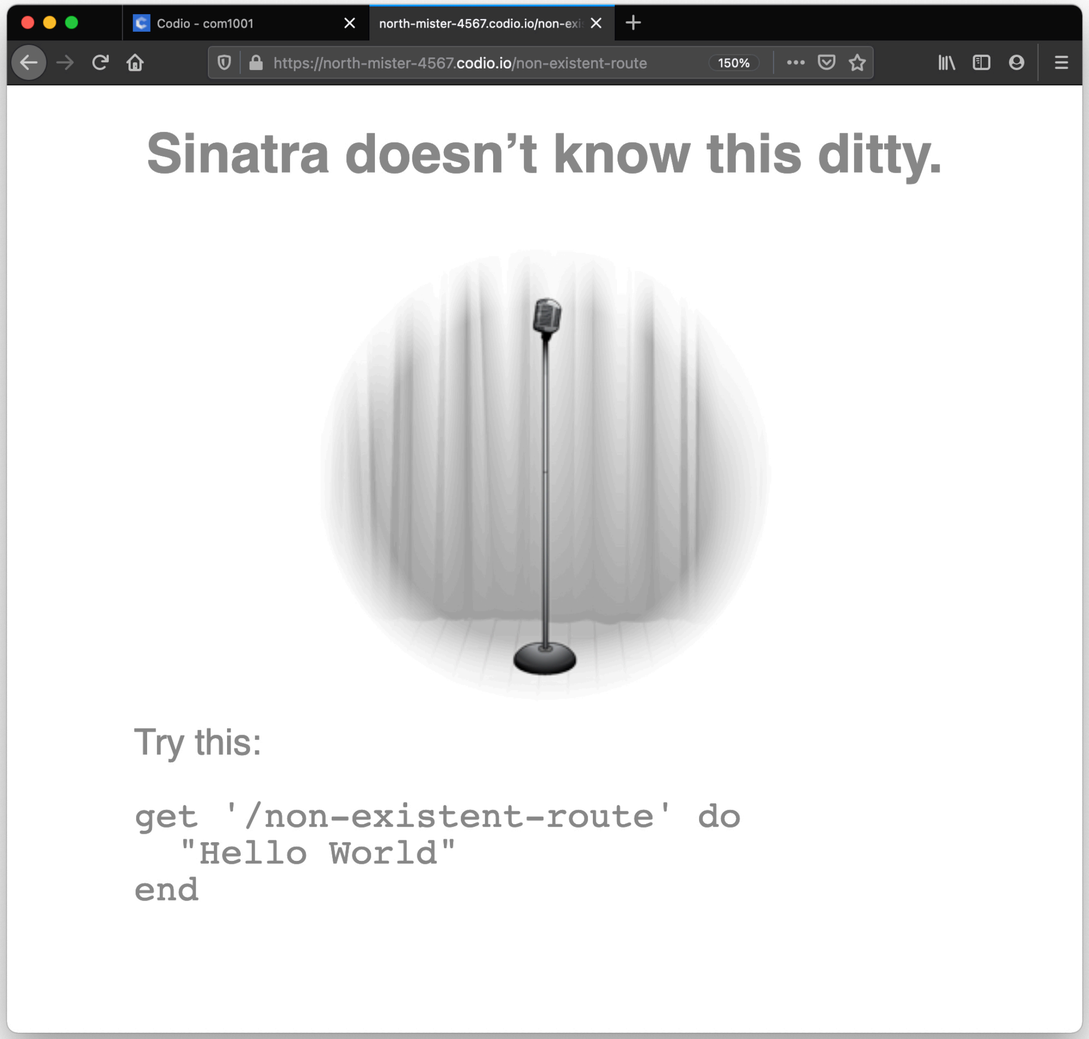
routes/greedy_wildcard.rb

When Sinatra parses routes, the first sufficient match found in the code is the one that will be executed.

This is true when there's a more specific route definition later in the controller.

This is what is displayed when we request `/introverted-route` with this example:

Remember the 404 Error?



You've probably seen this page plenty of times now.

In a production application, we'd want to replace this page with something else. We need the special `not_found` route:

```ruby
not_found do
  "Your custom 404 message or erb file goes here"
end
```

routes/handling_404.rb

Redirecting to 404

```ruby
def numeric?(str)
  str.match(/^(\d)+$/)
end

def date?(year, month, day)
  numeric?(year) &&
    numeric?(month) &&
    numeric?(day) &&
    Date.valid_date?(year.to_i, month.to_i, day.to_i)
end

get "/date/*-*-*" do
  y = params[:splat][0]
  m = params[:splat][1]
  d = params[:splat][2]

  not_found unless date?(y, m, d)

  "The date requested is #{y}-#{m}-#{d}"
end

not_found do
  "The page was not found"
end
```

routes/divert_to_404.rb

If the user gives us a bad input, we may not be able to do anything with it, and redirecting to 404 may be the only option.

This example builds on the earlier splat code. This time it validates the numbers supplied and sees if they represent a date. If they do not, the user is redirected to the 404 page.

When Things Go Wrong…

```ruby
if ENV["APP"] == "production"
  disable :raise_errors
  disable :show_exceptions

  error do
    "There was an error..."
  end
end

get '/' do
  numerator = 0
  denominator = 0
  numerator / denominator
end
```

routes/error_handling.rb

Unfortunately, web applications can fail in production, but we wouldn't want the usual diagnostic information Sinatra produces to be shown to the user.

This example disables errors and exceptions, and displays the result of the special `error` route.

Of course, we'd only want this enabled for web applications actually in **production** – in development mode, we want to see the diagnostic information.

How you determine whether your app is in production mode may be via some other means…