In this lecture, we will discuss…

- Query Parameters - GET
- POST Data
- Whitelisting parameters
- Cross Site Scripting (XSS)

HTTParty Client class

- Helper class - `app/services/movies_ws.rb`

```ruby
class MoviesWS
  include HTTParty
  base_uri "http://localhost:3000"
end

```

```
> MoviesWS.get("/movies/12345.json").parsed_response
 => {"id"=>"12345", "title"=>"rocky25", "created_at"=>nil, "updated_at"=>"2016-01-03T17:05:36.066Z"}
```

Parameter Types

- URI elements (e.g., :movie_id, :id)
- Query String - part of the URI, uses "?", and contains individual query parameters
- POST Data - in the payload body.

Parameter Types – Example

```
> MoviesWS.get("/movies.json?title=rocky25&foo=1&bar=2&baz=3").parsed_response
```

```
{"title"=>"rocky25", "foo"=>"1", "bar"=>"2", "baz"=>"3",
 "controller"=>"movies", "action"=>"index", "format"=>"json"}
```

Post Data – Example

```
MoviesWS.post("/movies.json",
     :body=>{:movie=>{:id=>"123457",:title=>"rocky27",:foo=>"bar"}}.to_json,
     :headers=>{"Content-Type"=>"application/json"})
```

```
{"movie"=>{"id"=>"123457", "title"=>"rocky27", "foo"=>"bar"},
  "controller"=>"movies", "action"=>"create", "format"=>"json"}
```

White Listing Parameters

- Rails has built in features based on parameters
- Controller has a “white list” of acceptable parameters
- White list with 2 fields

```ruby
def movie_params
  params.require(:movie).permit(:id,:title)
end
```

- Usage

```ruby
def create
  @movie = Movie.new(movie_params)
```

White Listing Parameters

```
{"movie"=>{"id"=>"123457", "title"=>"rocky27", "foo"=>"bar"},
  "controller"=>"movies", "action"=>"create", "format"=>"json"}
```

```
{"id"=>"123457", "title"=>"rocky27"}
```

Cross Site Scripting (XSS)

- Browsers can run scripts (JavaScript)
- If a user trusts a website, might allow the scripts to run
  - `<script type="text/javascript" > alert("Hard Disk Error. Click OK."); </script >`
- It is possible to inject malicious scripts into content from trusted sites
- Scripts can hijack user sessions, redirect user to other sites

Cross Site Scripting (XSS)

- ✧ POST request by default will fail
  - Can't verify CSRF (Cross Site Request Forgery) token authenticity - message
- ✧ Relax Security
  - `app/controllers/application_controller.rb`

```
class ApplicationController < ActionController::Base
  # Prevent CSRF attacks by raising an exception.
  # For APIs, you may want to use :null_session instead.
  #protect_from_forgery with: :exception
  protect_from_forgery with: :null_session
end
```

Query Parameters - Demo

Demo

Other Parameter - options

- Arrays

```
MoviesWS.get("/movies.json?id[]=12345&id[]=12346&foo[]=1&foo[]=2")
```

```
{"id"=>"12346", "foo"=>["1", "2"], "controller"=>"movies", "action"=>"index", "format"=>"json"}
```

Other Parameter - options

- Hash

```
key=>{"prop1"=>"val", "prop2"=>val}
```

```
MoviesWS.get("/movies.json?movie[id]=12345&movie[title]=rocky27&movie[year]=2016")
```

```
{"movie"=>{"id"=>"12345", "title"=>"rocky27", "year"=>"2016"},
   "controller"=>"movies", "action"=>"index", "format"=>"json"}
```

Summary

- ✧ Query Parameters – common way to request data
- ✧ CSRF security concerns and whitelisting parameters

What's Next?

- ✧ Methods