

# Files / CSV / ZIP

Devpoint Labs - Dave Jungst / Jake Sorce

# Reading / examining files

```
file = "my_file.txt"
```

```
File.read(file)
```

```
=> "This is the contents of my file\nThis is a new line"
```

```
File.exists? file
```

```
=> true
```

```
File.directory? file
```

```
=> false
```

```
File.dirname(file)
```

```
=> "."
```

```
File.basename(file, ".*")
```

```
=> "my_file"
```

```
File.atime file
```

```
=> 2015-10-07 14:44:14 -0600 (last time file was accessed)
```

```
File.extname file
```

```
=> ".txt"
```

# Writing / opening a file

You can create a new file or open a file with `File.open`. Using `File.open` in block form will ensure that the file is closed.

```
1 File.open("path/to/file/my_file.txt", "w") do |file|
2   begin
3     file << "#{content} \n"
4   rescue => e
5     logger.error e.messages
6   end
7 end
```

# File flags

`File.open('my_file.txt', 'r')` => Read only from beginning of file

`File.open('my_file.txt', 'r+')` => Read write from beginning of file

`File.open('my_file.txt', 'w')` => Write only overwrites existing or creates file

`File.open('my_file.txt', 'w+')` => Read / Write overwrites existing or creates file

`File.open('my_file.txt', 'a')` => Write only starts at end of existing file or creates new

`File.open('my_file.txt', 'a+')` => Read / Write appends or creates

`File.open('my_file.txt', '[r/w/a]b')` => Sets file to binary mode and encoding to ASCII



# Rails App for File Practice

```
rails new file_practice -d postgresql -T
```

```
gem 'haml-rails'
```

```
rails g controller dashboard index
```

```
rails g model user first_name last_name email phone
```

```
root 'dashboard#index'
```

```
bundle
```

```
bundle exec rake db:create db:migrate
```



# Writing files

Add a route to config/routes.rb for writing files

```
1 Rails.application.routes.draw do
2   root 'dashboard#index'
3   post 'dashboard', to: 'dashboard#write'
4 end
```

Create a view where a user can name a file and give the file some content through a text area

```
1 _# app/views/dashboard/index.html.haml
2 %h2.text-center File Practice
3 %hr
4 .col-md-10.col-md-offset-1
5   = form_for :file, url: write_file_path do |f|
6     .form-group
7       = f.label :filename, class: 'control-label'
8       = f.text_field :filename, {class: 'form-control'}
9     .form-group
10      = f.label :content, class: 'control-label'
11      = f.text_area :content, {class: 'form-control'}
12      = f.submit :create_file, class: 'btn btn-primary'
```

```
1 class DashboardController < ApplicationController
2   def index
3   end
4
5   def write
6     path = "#{File.expand_path('.')}/tmp/"
7     File.open("#{path}#{params[:file][:filename]}", 'wb') do |file|
8       begin
9         file.write(params[:file][:content])
10      rescue => e
11        logger.error e.messages
12      end
13    end
14
15    redirect_to :root
16  end
17 end
```

Create a write method in the controller. We are setting the path to the apps root\_dir/tmp.

The full filename is root\_dir/tmp/the\_file\_name

Setting the wb flag will create or overwrite an existing file

# Reading Files

For this example I want to read the contents of a file and display it on the screen.

I'm going to create a file called `my_file.txt` in `/tmp`. I'm going to populate that file with Lorem Ipsum text from a generator. What is Lorem Ipsum?

Then I will read the file in from a file input box and display a partial containing the text.



Lorem Ipsum is filler text commonly used in web design to pre populate areas of a web page for demo purposes. I use a Lorem Ipsum generator to generate my desired amount of paragraphs depending on the content. [www.lipsum.com](http://www.lipsum.com)

*"Neque porro quisquam est qui dolorem ipsum quia dolor sit amet, consectetur, adipisci velit..."*  
 "There is no one who loves pain itself, who seeks after it and wants to have it, simply because it is pain..."

**Lorem Ipsum** is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type and scrambled it to make a type specimen book. It has survived not only five centuries, but also the leap into electronic typesetting, remaining essentially unchanged. It was popularised in the 1960s with the release of Letraset sheets containing Lorem Ipsum passages, and more recently with desktop publishing software like Aldus PageMaker including versions of Lorem Ipsum.

Contrary to popular belief, Lorem Ipsum is not simply random text. It has roots in a piece of classical Latin literature from 45 BC, making it over 2000 years old. Richard McClinton, a Latin professor at Hampden-Sydney College in Virginia, looked up one of the more obscure Latin words, consectetur, from a Lorem Ipsum passage, and going through the cities of the word in classical literature, discovered the undoubtedly source, Lorem Ipsum comes from sections 1.10.32 and 1.10.33 of "de Finibus Bonorum et Malorum" (The Extremes of Good and Evil) by Cicero, written in 45 BC. This book is a treatise on the theory of ethics, very popular during the Renaissance. The first line of Lorem Ipsum, "Lorem ipsum dolor sit amet..", comes from a line in section 1.10.32.

The standard chunk of Lorem Ipsum used since the 1500s is reproduced below for those interested. Sections 1.10.32 and 1.10.33 from "de Finibus Bonorum et Malorum" by Cicero are also reproduced in their exact original form, accompanied by English versions from the 1914 translation by H. Rackham.

It is a long established fact that a reader will be distracted by the readable content of a page when looking at its layout. The point of using Lorem Ipsum is that it has a more-or-less normal distribution of letters, as opposed to using 'Content here, content here', making it look like readable English. Many desktop publishing packages and web page editors now use Lorem Ipsum as their default model text, and a search for 'lorem ipsum' will uncover many web sites still in their infancy. Various versions have evolved over the years, sometimes by accident, sometimes on purpose (injected humour and the like).

There are many variations of passages of Lorem Ipsum available, but the majority have suffered alteration in some form, by injected humour, or randomised words which don't look even slightly believable. If you are going to use a passage of Lorem Ipsum, you need to be sure there isn't anything embarrassing hidden in the middle of text. All the Lorem Ipsum generators on the Internet tend to repeat predefined chunks as necessary, making this the first true generator on the Internet. It uses a dictionary of over 200 Latin words, combined with a handful of model sentence structures, to generate Lorem Ipsum which looks reasonable. The generated Lorem Ipsum is therefore always free from repetition, injected humour, or non-characteristic words etc.

ected humour, or non-characteristic words etc.

☒ paragraphs

☐ words

☐ bytes

☐ lists

☒ Start with 'Lorem ipsum dolor sit amet...'

# Lorem Ipsum

*"Neque porro quisquam est qui dolorem ipsum quia dolor sit amet, consectetur, adipisci velit..."*  
 "There is no one who loves pain itself, who seeks after it and wants to have it, simply because it is pain..."

Lorum ipsum dolor sit amet, consectetur adipiscing elit. Quisque et eros elit. Morbi in est vestibulum, ornare sem ac, sodales nunc. Quisque id dui nec libero mollis efficitur efficitur vitae erat. Aenean non imperdiet enim. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos himenaeos. Nulla efficitur magna vel porttitor semper. Sed tellus risus, vulputate mattis dolor at, ultricies mollis lectus. Maecenas laoreet nibbi at lacinia molestie. Donec non tempus massa. Maecenas dapibus molestie nunc, eu tincidunt magna aliquam sed. Nunc consectetur turpis purus sit amet suscipit. Ut sagittis, felis sit amet fringilla tempus, tellus neque commodo orci, eu tincidunt eros justo a mi. Maecenas fringilla eget tortor nec pellentesque. Pellentesque aliquam auctor massa, sit amet luctus eros consectetur ornare. Integer accumsan lacus quis volutpat porta. Mauris finibus finibus congue.

Vivamus non eleifend lectus. Morbi rhoncus, magna a venenatis suscipit, lectus ante auctor eros, quis laoreet nibh metus ac nisi. Aliquam ac viverra elit. Morbi tincidunt odio eget hendrerit laoreet. Morbi finibus purus et imperdiet suscipit. Nunc convallis et massa a faucibus. Vestibulum nibh urna. vehicula posuere leo ac. dictum bibendum libero.

Aenean at erat ullamcorper, plaetra nibh in, imperdiet libero. Nam feugiat massa est, ut maleduora eore blandit eu. Fusce conat massa in neque congue, sit amet pharetra nisi aliquet. Suspendisse maleduora conwallis laoreet. Pellentesque at tempor nibh. Suspendisse conwallis purus vel lacus rutrum, vitae conwallis diam pellentesque. Aenean ac est id dui vitae faucibus nisi eget libero. Aenean ullamcorper odio vel neque ullamcorper, id viverra nisi laoreet. Sed pretium elit vitae varius posuere. Suspendisse faucibus semper consectetur. Curs scelerisque, lectus non dignissim ultrices, lectus eore maximus turpis, eget conwallis magna ligula eu mauris. Maecenas libero quam, pulvinar eget portitor imperdiet, tempusque qui elit. Donec sagittis rutrum tristique. Fusce augue tortor, iaculis sit amet tincidunt vel, bibendum ultrices nisi. Duis ac massa gravida, volutpat nulla eget, interdum felis. Donec eleifend purus id tellus bibendum, ornare dapibus nulla consequat.

Donec rutrum, odio vitae molestie condimentum, dolor ex ultrices orci, eget blandit mauris nunc quis mi. Proin sit amet iaculis ante. Quisque iaculis purus nisi, ut venenatis dui ornare ut. Nullam aliquet justo eu velit laoreet, ac viverra mauris interdum. Nullam ac dolor placerat, accumsan urna a, facilisis justo. Aliquam vestibulum massa dapibus tincidunt imperdiet. Integer non magna ut diam convallis malesuada eu vitae felis.

Aliquam sed eros ut diam blandit gravida non eget mauris. Aliquam utricles elit ac felis facilisis blandit a nec justo. Vestibulum sagittis, mi sed euismod pellentesque, ante velit finibus massa, a feugiat justo elit vel felis. Curabitur aliquam quis odio nec gravida. Aenean lobortis, dolor vel sodales laoreet, augue justo convallis dui, a mollis odio ligula non ex. Nulla malesuada ut eros id venenatis. In id rutrum nisi. Sed interdum neque in libero egestas porta. Quisque commodo a eros ac efficitur. In consequat non ante id mattis. Phasellus fermentum maximus nisi in fermentum.

Generated 5 paragraphs, 480 words, 3254 bytes of Lorem Ipsum



# Writing files

Add a route to config/routes.rb for reading files

```
1 Rails.application.routes.draw do
2   root 'dashboard#index'
3   post 'write_file', to: 'dashboard#write'
4   post 'read_file', to: 'dashboard#read'
5 end
```

Add the ability for a user to upload a file to  
The server using f.file\_field

```
13 %hr
14 .col-md-10.col-md-offset-1
15 = form_for :file, url: read_file_path, action: 'read' do |f|
16   .form-group
17     = f.file_field :uploaded_file, {class: 'form-control'}
18   .form-group
19     = f.submit :read_file, class: 'btn btn-primary'
```

```
1 -# app/views/dashboard/_content.html.haml
2 = link_to "Back", :back
3 %p
4   = simple_format(@content)
```

Create a partial in app/views/dashboard/\_content.html.haml where we can render the contents of the file and link the user back.

The simple\_format method will convert new lines to <p> in HTML

# Read controller

```
18 def read
19   File.open(params[:file][:uploaded_file].tempfile.path, 'r') do |file|
20     begin
21       @content = file.read
22       rescue => e
23         logger.error e.messages
24       end
25     end
26
27     render partial: 'content'
28   end
29 end
```

When a file is uploaded to Rails a Tempfile file object is created. Notice the .tempfile method being called.

Alternatively the entire block could have been written:

```
File.open(params[:file][:uploaded_file].tempfile).read
```

# CSV

CSV's are similar to files although we can pass in options and we can read the CSV 1 row at a time to create objects in memory.

CSV comes installed with Rails but you still need to require it in controllers that will use it

```
app/controllers/dashboard_controller.rb
```

```
require 'csv'
```



# Reading / Parsing a CSV

I have created a CSV in tmp called users.csv

```
1 first_name,last_name,email
2 'Dave','Jungst','dave@fakeremail.com'
3 'Jake','Sorce','jake@fakeremail.com'
4 'Bob','Bobson','bob@fakeremail.com'
5 'Abe','Lincoln','abe@fakeremail.com'
6 'Spider','Man','spider@fakeremail.com'
7 'Bat','Man','bat@fakeremail.com'
```

A CSV is just comma separated values which use new lines as rows

The first line is assumed to be the column headers

# Parsing a CSV

```
CSV.parse(File.read('tmp/users.csv'))
```


```
=> [["first_name", "last_name", "email"],  
     ["'Dave'", "'Jungst'", "'dave@fakeremail.com'"],  
     ["'Jake'", "'Sorce'", "'jake@fakeremail.com'"],  
     ["'Bob'", "'Bobson'", "'bob@fakeremail.com'"],  
     ["'Abe'", "'Lincoln'", "'abe@fakeremail.com'"],  
     ["'Spider'", "'Man'", "'spider@fakeremail.com'"],  
     ["'Bat'", "'Man'", "'bat@fakeremail.com'"]]
```

This returns an array of arrays each inner array is a row from the CSV the first row containing the column headers.

This can also be done like: `CSV.read("tmp/users.csv")`

# Building an object from a csv

```
users = []  
header = true  
CSV.foreach('tmp/users.csv') do |row|  
  if header  
    header = false  
  next  
end  
users << { first_name: row[0], last_name: row[1], email: row[2]}  
end
```



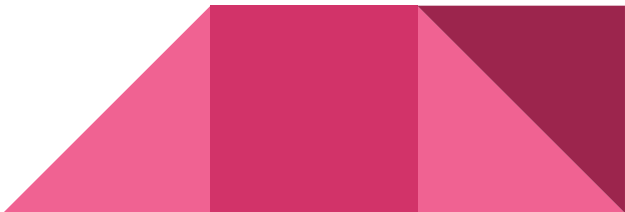
# Better

```
users = []  
CSV.foreach('tmp/users.csv', headers: true) do |row|  
  users << { first_name: row['first_name'], last_name: row['last_name'], email: row['email'] }  
end  
  
#<CSV::Row "first_name":"Dave" "last_name":"Jungst" "email":"dave@fakeremail.com">
```



# Best

```
users = []  
CSV.foreach('tmp/users.csv', headers: true, header_converters: :symbol) do |row|  
  users << { first_name: row[:first_name], last_name: row[:last_name], email: row[:email] }  
end  
  
#<CSV::Row first_name:"Dave" last_name:"Jungst" email:"dave@fakeremail.com">
```





# Writing to a CSV

```
4 CSV.open("my_file.csv", "wb") do |csv|
5   csv << ['this', 'is', 'a', 'row']
6   csv << ['this', 'is', 'another', 'row']
7 end
```

Pass a path/file to the CSV

Pass a flag (wb) in this case replace or create a new file

The block argument |csv| is the I/O stream

Push arrays on to the CSV as rows



# RubyZip

RubyZip is a gem that is most commonly used for compressing and decompressing files in ruby.

```
gem 'rubyzip'
```

```
require 'zip'
```

<https://github.com/rubyzip/rubyzip>



# Project

1. Create a rails app
2. Create a User model with at least first\_name, last\_name, email
3. Create users by uploading a csv of users
4. Create files containing user's info from the CSV (plain text)
  - a. -First Name: The user's name
  - b. -Last Name: The user's last name
  - c. -Email: The user's email

## BONUS:

1. Append Lorem Ipsum to the user's files read in from another file
2. Add unique to email and don't create users that already exist
3. Read in all user's files in a block to create a master file
4. Upload a Zip of CSV's that is then decompressed and used to create users