OFFICE HOURS

SIGN OUT

Successfully signed in!

LESSON 3: COLLECTIONS & LOOPS

| Slides | Materials | Lecture | Exit Ticket | Instructor Notes |

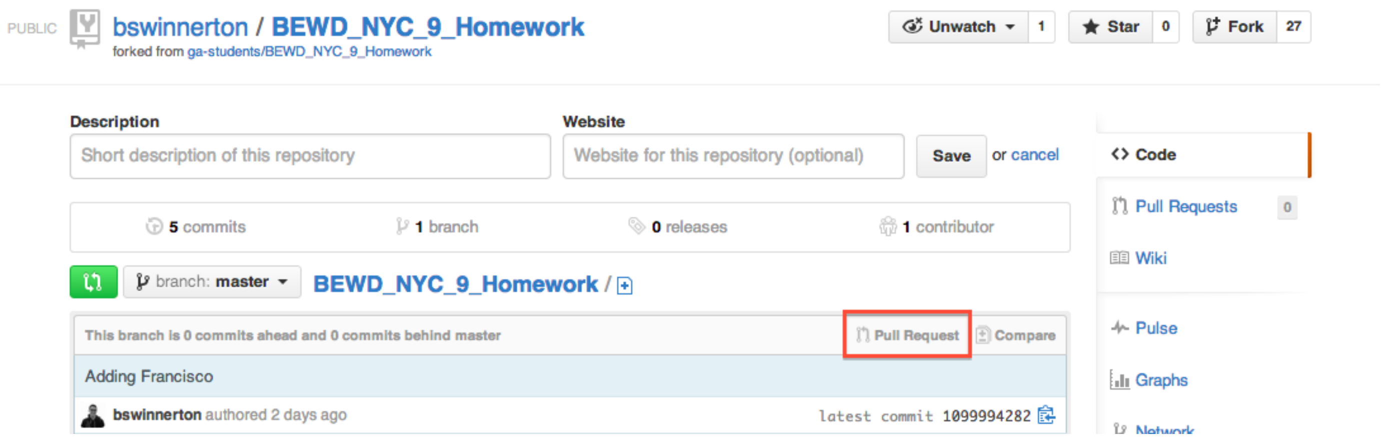
Description:

Looping, Arrays & Hashes

Classwork: (Answer)

Submitting your Secret Number homework

We will be using Pull Requests in order to "submit" your homework assignments. In order to do this, you will need to `git push origin master` (the last step, here). Once you do, you should see the updated code in a URL similar to https://github.com/your-username-here/secret_number. Visit that page and click the "Pull Request" button.


PUBLIC bswinnerton / BEWD_NYC_9_Homework
forked from ga-students/BEWD_NYC_9_Homework

Description: Short description of this repository
Website: Website for this repository (optional)

5 commits · 1 branch · 0 releases · 1 contributor

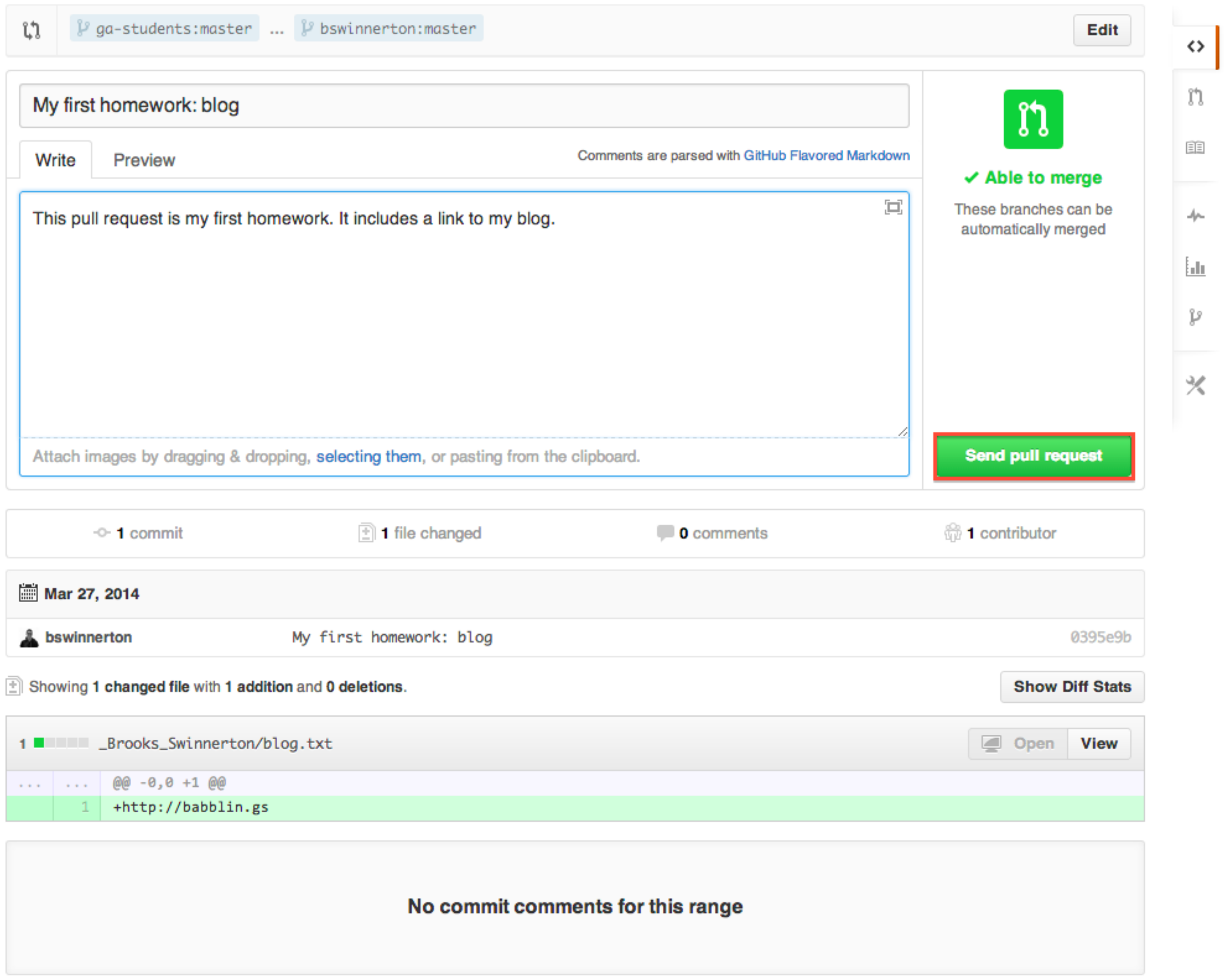
branch: master · BEWD_NYC_9_Homework / +

This branch is 0 commits ahead and 0 commits behind master

Adding Francisco

bswinnerton authored 2 days ago · latest commit 1099994282

On the following page, you will see a place to enter a message, and a large green button that says "Create Pull Request". If you have any notes, or specific pieces to your homework that you would like us to take a look at, be sure to add those to the message. Then, click "Create Pull Request".


ga-students:master ... bswinnerton:master

My first homework: blog

This pull request is my first homework. It includes a link to my blog.

Comments are parsed with GitHub Flavored Markdown

Attach images by dragging & dropping, selecting them, or pasting from the clipboard.

✔ Able to merge
These branches can be automatically merged

1 commit · 1 file changed · 0 comments · 1 contributor

Mar 27, 2014
bswinnerton My first homework: blog 0395e9b

Showing 1 changed file with 1 addition and 0 deletions.

```
1 _Brooks_Swinnerton/blog.txt
@@ -0,0 +1 @@
+http://babblin.gs
```

No commit comments for this range

That's it! We will get back to you as soon as we can with comments and review.

Iteration Exercise

First, fork this repository and run:

```
cd ~/Sites/
git clone https://github.com/your-username-here/bottles_of_beer.git
```

Next, copy the following code into sublime text, and save it as `~/Sites/bottles_of_beer/your_name_here/bottles_of_beer.rb`:

```
# 99 Bottles of Beer on the Wall Exercise
#
# Write a program that prints 99 bottles of beer on the wall. The song goes a
# so:
#
# 99 bottles of beer on the wall
# 99 bottles of beer!
# You take one down and pass it around,
# 98 bottles of beer on the wall!
# ...
#
# And ends with
# 1 bottle of beer on the wall
# 1 bottle of beer!
# You take one down and pass it around,
# No more bottles of beer on the wall
```

Follow the instructions in the comments, and when you are finished, save the file, commit the code to Git and push it to GitHub:

```
cd ~/Sites/bottles_of_beer/
git add your_name_here/bottles_of_beer.rb
git commit -m "Adding bottles_of_beer exercise"
git push origin master
```

Array Exercise

First, fork this repository and run:

```
cd ~/Sites/
git clone https://github.com/your-username-here/arrays.git
```

Next, copy the following code into sublime text, and save it as `~/Sites/arrays/your_name_here/arrays.rb`:

```
# Array Exercise
#
# Edit the contents of `top_right`, `bottom_center` and `center`
# according to the comments in each of the methods. Once completed, run this
# file from your terminal (remember, we use the `ruby` command), and you shou
# see a success or failure message for each.
#

matrix = [
  [0, 1, 2],
  [3, 4, 5],
  [6, 7, 8]
]

def top_right(matrix)
  # This method should return (not puts) the number all the way to the top
  # right of the variable `matrix`
end

def bottom_center(matrix)
  # This method should return (not puts) the number all the way to the bottom
  # center of the variable `matrix`
end

def center(matrix)
  # This method should return (not puts) the array in the middle of the `matr
end

if top_right(matrix) == 2
  puts 'SUCCESS: You got the top right element!'
else
  puts "It looks like the `top_right` method isn't returning the right value"
```

```
end

if bottom_center(matrix) == 7
  puts 'SUCCESS: You got the bottom center element!'
else
  puts "It looks like the `bottom_center` method isn't returning the right va
end

if center(matrix) == [3, 4, 5]
  puts 'SUCCESS: Congratulations, you finished the exercise'
else
  puts "It looks like your `center` method isn't reutrning the right value"
end
```

Follow the instructions in the comments, and when you are finished, save the file, commit the code to Git and push it to GitHub:

```
cd ~/Sites/arrays/
git add your_name_here/arrays.rb
git commit -m "Adding array exercise"
git push origin master
```

Homework:

Optional Exercise

Below is an extra, optional homework assignment to test your knowledge on navigating data structures and methods.

First, fork this repository and run:

```
cd ~/Sites/
git clone https://github.com/your-username-here/radiolab_blog_to_html.git
```

Next, copy the following code into sublime text, and save it as

~/Sites/radiolab_blog_to_html/your_name_here/radiolab_blog_to_html.rb:

```
# RadioLab Blog
#
# The purpose of this exercise is to utilize the skills that you've learned
# to access and retrieve values from arrays and hashes to create an HTML page
# that your browser can open.
#
# Below is an array of blog posts, which contains three individual posts,
# stored as hashes. Each hash has four attributes: the title, body, a child
# hash (which contains the authors first name, last name and twitter handle),
# and a child array which contains various tags about the post.
#
# Your job is to fill in the logic at the end of the `blog_posts` array,
# following the comments as your guide.

blog_posts = [
  {
    title: 'What Makes A Star Starry? Is It Me?',
    body: "Notice what Tyler Nordgren does in these posters. He's an artist,
    author: {
      first_name: 'Robert',
      last_name: 'Krulwich',
      twitter_handle: '@rkrulwich'
    },
    tags: ['art', 'education', 'science']
  },
  {
    title: 'The Meter: The Measure of a Man',
    body: "About six and a half billion people use the metric system every si
    author: {
      first_name: 'Latif',
      last_name: 'Nasser',
      twitter_handle: '@latifnasser'
    },
    tags: ['discovery', 'dialogues', 'history', 'meter', 'science', 'storytel
  },
  {
    title: 'Shattering Silence and An Eye of God',
    body: "In our Morality show, we tell the story of Eastern State Penitenti
    author: {
      first_name: 'Brenna',
      last_name: 'Farrell',
      twitter_handle: '@brennacfarrell'
    },
    tags: ['history', 'morality', 'prison']
  }
]

## HTML conversion methods - there should be no need to edit any of these

# This method will requires one parameter to be passed to it (referenced as
# `title` everywhere inside the method) and surround it with <h1> (header) ta
def title_to_html(title)
  "<h1>#{title}</h1>"
end

# This method requires three parameters to be passed to it. The first paramet
# should be the first name of the author, the second parameter should be the
# last name of the author, and the last parameter should be their twitter
# handle. It will then output a string that is italicized and linked to their
# twitter page.
def author_to_html(first_name, last_name, twitter_handle)
  "<p><i>Author: <a href='http://twitter.com/#{twitter_handle}'>#{first_name}
end

# This method requires one parameter to be passed to it (referenced inside of
# the method as `body`, and it will return to you the body surrounded with <p
# tags.
def body_to_html(body)
  "<p>#{body}</p>"
end

# This method uses some Array sorcery in order to take multiple elements of a
# array and output one sentence with commas separating each element.
def tags_to_html(tags)
  tags.join(', ')
end

## End HTML conversion methods

# Starting here we're going to build a string that is the start of the HTML
# that we want to be displayed in our browser. This string, `html`, contains
# the title, and the start of the body.
html = '<html><head><title>BEWD RadioLab Blog</title></head><body>'
```

```
# Create a loop that iterates through each of the blog posts. Since
# `blog_posts` is an array, you can start at 0 and loop to the size of the
# elements in the array. Make sure that you have a way to access the current
# iteration number when looping over the array.


  # Once inside of the loop, create a new variable called `post` (singular)
  # that is equal to the value of the `blog_posts` array at the current
  # iteration number as the index. Don't forget, indexes start at 0, so you m
  # need to offset the current iteration number first.


  # Now that we have access to a single element of `blog_posts` (which you
  # you stored in a variable called `post` above) we need to extract the
  # :title key from that post hash, and append it to the `html` string that w
  # defined out of the loop. You can append to strings in the same way that y
  # add something to an array, with `<<`. For example 'Brooks <3s ' << 'cats'
  # will return 'Brooks <3s cats'. Also, since we have a method that can help
  # convert the title to html, defined above, be sure to pass the title for t
  # current post to that method and append the result to the `html` variable.


  # You will want to do the same thing for the author. But remember, the meth
  # that is defined above takes in three parameters, the first name, last nam
  # and twitter handle. Once you have executed that method, append the result
  # to `html`.


  # Same idea again here for the body of the current post that we're iteratin
  # over. Pass the body of the current hash to the html helper method definec
  # above and then append the output to the `html` variable.


  # Lastly, do the same thing for the tags of the `post` (run the appropriate
  # helper method defined above and append the output to the `html` variable.
  # Since the tags are stored as an array, you would think that we need to
  # somehow iterate over it, but instead we leave that job up to the method
  # defined above that converts tags them to HTML.


# Be sure to end your loop here.
# Finally, we use this line to append the closing tags to our HTML
html << '</body></html>'

# And here, we use some Ruby magic to write the contents of the `html` variak
# to a file named blog.html, and then we use the %x method to tell our comput
# to open the file in our default browser. You should see the following page
# display in your browser if you successfully completed the exercise:
# http://i.imgur.com/6GmCNVU.png
File.write('blog.html', html)
%x(open 'blog.html')
```

Once completed, you can run the file by typing `ruby navigating_file_structures.rb` , and it should automatically open your browser. You should see a page that looks similar to this if you successfully completed the assignment.

If you feel comfortable with the assignment, save the file, commit the code to Git and push it to GitHub:

```
cd ~/Sites/radiolab_blog_to_html/
git add your_name_here/radiolab_blog_to_html.rb
git commit -m "Adding data structure optional homework"
git push origin master
```

Resources:

Practice your ruby skills with exercises from Rubeque. You can search the problems by tags like "array", "method", etc.