Girl Develop It

don't be shy. develop it.

TEACHER HANDBOOK

First, and most importantly, thank you for offering to teach!

Here are a list of expectations and guidelines we share with all our new teachers. It's a long, informative document so please read it all the way through. These guidelines are meant as a resource for you, so if you have feedback or additions that would make them better, please add a comment below or send an email to the organizer.

Organizer contact information

Mina Markham, Chapter Leader - mina@girldevelopit.com, 316-655-1671, @minamarkham
Leslie Wilson, Chapter Leader - leslie@girldevelopit.com, 214-444-9537, @lesliewdotcom

About Girl Develop It

Girl Develop It (GDI) launched in 2010 with a simple concept, make software development accessible and affordable to all women in New York City with little to no development or training experience. Founded by female software developers Sara Chipps and Vanessa Hurst, the organization currently has 24 chapters located across North America, from Austin to Boulder to Boston, Canada, and Australia.

GDI's official mission statement

- GDI exists to provide affordable and accessible programs to women who want to learn software development through mentorship and hands-on instruction.
- Make sure women of all ages, races, education levels, income, and upbringing can build confidence in their skill set to develop web and mobile applications.

Class goals: To demystify programming; to build confidence and empower students; to help teach students to *think* like programmers; to emphasize that:

- It's okay to **not** be an expert. Advanced programmers are still learning all the time!
- Coding is not about memorizing. It's more important to know where to find resources and help online and in the local tech community.
- It's okay to not understand everything immediately. Keep learning and experimenting and coding knowledge starts to snowball.
- Errors are equally powerful when learning to code.
- There are always multiple solutions to a given coding problem.

We're not superhuman and we (unfortunately) can't create a fleet of expert female coders in four to six weeks. While learning is definitely a goal for classes, empowerment is equally, if not even more, important. Ideally, we want students to be pumped enough to keep coding long after class is over. If students leave

feeling excited about coding, then: mission accomplished.

The bottom line: We are all about providing a friendly, open environment where women can have fun coding.

Class Structure and Topics

Our focus is web and mobile development. Topics taught by other chapters include introductory JavaScript, Programming (in Ruby), and HTML/CSS, among others.

Core curriculum is provided by Girl Develop It and is available on Github. Some slide decks are also posted on the website's "Resources" page. Visit either location to see current course material availability.

Teachers can tweak core curriculum to add additional lab exercises, clarify confusing slides, and so on. If you end up making tweaks, make sure to send your deck to an organizer so that we have your slides for future classes as well. Be sure any changes fit the level of your class - sometimes certain points are excluded from class slides because we're trying to keep things at the beginner level. Also be sure to update your TAs on any changes so they can be prepared for class.

We can also optionally create our own class materials and have them vetted by Girl Develop It headquarters, or acquire class materials from other Girl Develop It chapters. If you have something new you'd like to teach, let an organizer know and we'll work with you to try to make it happen!

See the tips below on Curriculum Creation.

Class Structure

HTML/CSS classes generally consist of four 2-hour sessions spread over four weeks.

Programming classes tend to be more challenging for students, so we expand programming classes to six classes over six weeks. Note that the curriculum will consist of four slide decks which we then spread out over the six classes.

Completing the curriculum is less important than empowering students - don't be worried if you're not progressing through the material as fast as the core curriculum specifies. If you're on class 2 material at class 4 but your students feel empowered, your class has been a success.

That said, if you find your class was too fast or too slow, feel free to suggest a different length/format for future rounds of classes.

Teaching Assistants

TAs do not need to be experts! We encourage current and past students to come back and TA our classes. It's a great way to solidify knowledge after taking a class.

We especially look for female developers, students, and new coders to TA classes, but we are also always excited to have male TAs who are passionate about GDI's mission. We try to have a higher ratio of female to male TAs so we're giving women the opportunity to have leadership roles and actively promoting our

mission, as well as, providing positive role models.

Currently, we do not limit the number of TAs per class. We've found that a higher TA-to-student ratio provides more support to our students, especially in programming classes. Some classes may have close to a 1:1 TA-to-student ratio.

Choosing TAs for your class

Teachers are not responsible for finding their own TAs currently. Organizers will recruit TAs for each class and provide their names and contact information to teachers.

If you know someone who you'd like to have TA, feel free to contact them directly and then send their information to the organizers. We keep a Google Drive spreadsheet of TA information for each class round. Alternatively, if you have access to the spreadsheet, you can add TA contact information to the spreadsheet directly. (You can request access, if you don't already have it, from an organizer.) Please make sure you're keeping a higher female-to-male TA ratio!

Meeting with your TAs before class

Because there may be many TAs present in your class, we encourage you to reach out to TAs before the class starts to establish guidelines for the class. For example, do you want TAs jumping in to add material during the lecture? Or is your teaching style more to answer questions during the lecture portion yourself and have TAs jump in when explicitly asked?

Other TA expectations you might address:

- If TAs cannot attend all classes, they can be floaters. We ask that TAs let the teacher know ahead of time whether they will be attending all classes or TAing on a floating basis.
- We encourage TAs to view the slides prior to class.
- There are generally 3 or more coding labs per class. TAs should stand up, walk around, and make themselves available during these times to answer student questions.
- Some students may feel less comfortable asking directly for help. If a student looks lost or has been mostly quiet, ask her how she's doing.
- Be careful, especially in Newbie classes, with bringing up extra material that has not been covered in the slides; teachers may be leaving out some information during class to avoid confusing or overloading students. If you're not sure whether to bring a topic up, ask the teachers: "Would this be a good time to address _____ or should we wait until later in the class?"

Laptops and Class Setup

Students are asked to bring their own laptops, but we have access to a computer lab in necessary. We ask that students inform us ahead of time if they will need a laptop for class. Alternatively, students without laptops can also pair with other students.

We are not OS-specific - either a Mac or a PC will do - and there are no real requirements as far as what computer students should bring. Be aware that sometimes students bring work computers that do not have admin rights! If a student is experiencing install errors during the first class, this may be the problem.

Text Editors and Environment Setup

A list of necessary class setup is usually included in the core curriculum slides. For text editors, we

generally recommend:

Sublime Text (Mac & Windows)
Notepad++ (Windows)

Teachers can have students install programs during the first class. For classes that require more setup (e.g. PHP), consider sending out an email to students prior to class with install instructions. You can also offer to help students with installation at a Code and Coffee and/or Hack Night prior to the class start date.

Regardless, there will likely always be last-minute class RSVPs or students that miss the installation instructions. Providing them earlier will at least minimize the amount of setup time needed in the first class.

Class Attendance and Cancellation Policies

Students are encouraged to attend all classes. If a student knows up-front that they won't be able to make a class, we suggest they wait until the next round, especially for the shorter HTML/CSS classes.

Life happens, and sometimes students have to miss a class. In those cases, students are welcome to attend a Hack Night, Code and Coffee, or similar to ask for additional help catching up. Teachers can also optionally offer virtual office hours through Google Hangout, Skype, or http://ohours.org/ (but this is entirely up to the teacher and definitely not required).

Class Cancellations and Teacher Attendance

Teachers must be available for all classes! If an emergency comes up, work with the organizers so we can support you and help with rescheduling your class. If you know ahead of time that the scheduled class times won't work with your schedule, talk to the organizers as early as possible. We've found that skipping a week of class affects class retention and adds to confusion when students are learning more complex programming topics.

If you're co-teaching a class there's a little more leeway here. If you know ahead of time that you will have to miss a class, let your co-teacher and the organizers know before the class start date! We may not reschedule in that case if your co-teacher is okay with flying solo one week.

Once scheduled, classes should not be cancelled except for in emergencies and definitely not without talking to an organizer first! Please keep us informed so we can help make classes awesome for everyone.

Acquiring Student Contact Info and Feedback

Meetup doesn't provide us with students' email addresses, so teachers should acquire contact information on the first day of class (using a Google Document, survey form, or other means).

Survey templates used for previous classes are available and can be adapted for your class. Some classes have provided optional, anonymous surveys after each class to provide students the opportunity to feedback that a class was too slow/fast, which topics were over their heads, and so on.

We also have used surveys at class beginning and end to acquire overall class feedback: How did the student find out about GDIDFW/the class? What skills did she have coming into the class? Does she feel confident in her grasp of the class concepts? Etc.

Currently, it's completely up to the teacher to decide how they'd like to acquire feedback. At some point, we might standardize surveys and survey guidelines a bit more. If you'd like to use surveys, talk to the organizers and we can provide you with past survey examples and other helpful information.

Gender Ratio

Male students are welcome to attend classes.

Homework and After-Class Resources

Teachers can decide what types of out-of-class exercises or resources (if any) they'd like to provide to students. In the past, teachers have used homework and/or provided additional learning resources in post-class emails.

Many students like having resources to refer to between classes, so after-class emails can be helpful. The emails might include: videos that expand on class topics or address them in a different way, links to resources like Codecademy, book recommendations, class clarifications or notes, and/or a link to that day's class slides. You can view example follow-up emails from past classes below.

Some teachers (especially of programming classes) have found providing homework beneficial. Homework lets teachers know which concepts students are getting stuck on.

If you choose the homework route, keep in mind that many students have full-time jobs and outside commitments. Keep the assignments relatively short and avoid overly complicated coding homework. Homework should be helpful to students, not frustrating or overwhelming.

We recommend letting students know how long the homework assignment should take - e.g., "if this assignment is taking longer than ___, ask for help!" You might also have answers available to students in case they get stuck.

Other ideas and advice from past teachers:

- Break down assignments into smaller steps with instructions for each step
- Providing virtual office hours (or asking if any of your TAs would be interested in running virtual office hours) might also be helpful if students are getting stuck often.
- Consider using bit.ly to track whether students are using resources, so you have a better idea of what students are finding interesting/helpful.
- Hack Nights and Code and Coffees are also great venues for getting help with homework or class concepts!

Resources (this is just the beginnings of a list - please add to it!)

- Cloud 9 IDE - https://c9.io/
- Google Hangout
- Ohours - http://ohours.org
- JS Fiddle - http://jsfiddle.net/
- Repl.it - http://repl.it/
- Dabblet - http://dabblet.com/

Class Outreach

Currently, organizers take care of outreach for classes. That said, if you have ideas, we want to hear them! We can use all the help we can get, so we would super happy if you would help promote classes to your social media network, friends, family, and so on.

Scholarships

We are working to make scholarships available for students. Potential students interested in scholarship opportunities should get in contact with an organizer prior to class. We will generally have about 2 scholarships available per class, so it's better to contact an organizer sooner rather than later.

After we receive an email, we'll forward the potential student the scholarship application form and determine scholarship eligibility from there.

Teacher Payments

Teachers should generally receive payment from Girl Develop It headquarters a week after classes end. Feel free to follow up with an organizer if you haven't received your payment!

Teacher payments consist of 50% of class tuition. If two teachers are co-teaching a class, each will receive 25%.

Wow, that's a lot of information. If you have any questions, feel free to contact your chapter leaders.

Thank you again for your time!

Mina Markham
Founder & Co-Organizer
mina@girldevelopit.com

Leslie Wilson-Wendling
Co-Organizer
leslie@girldevelopit.com

Curriculum Creation

- Read Confessions of a Public Speaker. It's got great tips and tricks in it for speakers and teachers alike.

Making the materials:

- Keep slides as visual as possible, not text-based. The majority of the slide should be taken up by either a short code snippet, a diagram, a screenshot, or in rare cases, a short list. Since you are teaching in person, you will be providing the textual narration needed, so you needn't overwhelm the slides with text. As a compromise, you could add a button to your slides for people to toggle the text/notes on after, while they're reviewing at home.
- Try to use visual, fun, local examples in your slides, in-class examples, and exercises, so that the audience has something to connect to. Animals, childhood loves, sports teams, TV shows. Anything "retro" works great, because everyone loves to get nostalgic.
- For hands-on workshops, make the exercises for each concept build from simple to complex. When in doubt, start with something you'd almost consider "too simple" - i.e. something that is 95% the same as a snippet that's already in the slides. Then add additional exercises or marked "bonus" sections in an exercise. It's important to communicate to students that they aren't expected to finish all the exercises, but it's also important to have enough exercises so that the more advanced students feel challenged.
- If you plan to do any live coding to demonstrate a concept, prepare it ahead of time. Print it out so you can reference it during the workshop, but try to roughly memorize what it is you're going to do. Or, prepare a file with parts commented out, if that's less error prone for you. Just do not expect to be able to live code and come up with a great example of a concept on the spot. Live coding is hard enough.

Creating Exercises:

- If the student is meant to start with some code instead of write from scratch, there should be a clear link to that and explicit instruction to "save file into your project folder". (In some of the Node exercises, it'd say "make a server like X", and we weren't sure what "like" meant).
- The first step should just be there to make sure that the student is set up in the environment and has got the most basic thing working. For example, they might be instructed to change the value of a string and reload.
- The subsequent steps should ramp up in difficulty.
    - To make it easier for students, you can link to relevant documentation, like: "Write the form data to a text file using <a>fs.readFile"
    - You can also make it more clear by saying where they should be making a change, like "in the makeBody function, do X."
    - You can also suggest that they check the slides like "for a reminder of how to use X, check slides Y-Z".
    - The degree of hints that you give really depend on what it is you're trying to make sure they learn, but I think we aim towards providing more hints than less in our curriculum, because we want to reduce confusion while still encouraging learning.
- The exercise should conclude with "Bonus" steps, for the more advanced students that find themselves with more time.

Delivering the materials:

- Practice your materials. That doesn't mean you need to say them aloud and make a mini audience of stuffed animals, but you should atleast be skimming through your slides and going over what you'd say in your head. What you're trying to figure out by practicing is whether there are any parts that don't jive - if there's an awkward transition, or something that needs a different example, or concepts out of order - and then fix the issue by figuring out the explanation in your head or changing the slides. Ideally, you feel comfortable enough with the material and it flows well enough that you don't feel like you need to memorize it, you can simply follow along with the story outlined by your slides.
- You can record speaker notes as a tool while practicing your content, but it doesn't usually work well to actually reference notes while presenting. If you're looking at notes while teaching, your students won't feel like you're connecting with them and will feel less engaged. You want to be looking at them at all times, when possible. Sure, you might miss covering some things if you don't reference your notes, but it's better to do that than to lose your audience entirely.
- Speak with energy and enthusiasm in your voice. If you sound bored or tired, then your audience will pick that up from. If you sound excited, then your audience will think they should be excited too. This is *particularly* important when you're teaching technical material over long periods of time, because it is easy for your audience to drift away. You have to work extra hard to keep them engaged.

Post-Event Follow-Up Email

Thanks for attending the workshop! The slides are available at:

[URL]

We will try to always keep them available at that URL.

We'd love your feedback on the workshop. If you have a few minutes, please provide it here (or just reply to this email):

[URL- example spreadsheetsform:

https://docs.google.com/spreadsheet/ccc?key=0Ah0xU81penP1dHVXYWZSZHM5VktNSndXenFrMXRjZ3c]

To continue your learning, here are some recommended resources:

Future GDI workshops:

[URLs]

Online tutorials:

[URLs]

Books:

[URLs]

Blogs:

[URLs]

Hope to see you at future meetups!

Sample follow-up email after HTML/CSS:

Hi all!

Thanks for coming out yesterday. The slides and exercises will remain available at:

https://dl.dropbox.com/u/10998095/htmlcss-nov4/index.html

We didn't do the embed exercise or the final layout exercise in class, so you can work on those now if you'd like.

Future Classes

Now that you've learnt the basics of HTML and CSS, you might want to take a JavaScript class next. Our JS workshop next weekend is full, but there's noone on the waiting list, so there's a chance you'll make it in if

someone cancels: http://www.meetup.com/Girl-Develop-It-San-Francisco/events/85864652/
We also have a CSS3 workshop next week on Thursday, and it still has spots left:
http://www.meetup.com/Girl-Develop-It-San-Francisco/events/87378692/

Here are some online resources:

Keep learning!

Lynda

Code School

Codecademy

Treehouse

Coding for Good

Blogs to keep up your web development skills:

CSS-Tricks

Smashing Magazine

A List Apart

HTML5 Weekly

Get Inspired by great web designs:

Web Creme

The Best Designs

Media Queries

Let me know if you have any questions. :-)
- pamela

Sample follow-up email after JavaScript class:

Thanks for attending the workshop! The slides are available at:
http://tinyurl.com/gdisf-javascript-intro
We will try to always keep them available at that URL.

We'd love your feedback on the workshop. If you have a few minutes, please provide it here (or just reply to this email):

https://docs.google.com/spreadsheet/viewform?formkey=dG5JSlZfaVhNaTlSVmU0S0lHd3NiZUE6MQ

To continue your learning, here are some recommended resources:

These workshops are designed to follow Intro to Javascript, in order so if you'd like to continue your journey:

- Advanced JS: Intro to jQuery
- Advanced JS: AJAX Workshop
- Advanced JS: Working with APIs
- Advanced JS: Intro to MVC Frameworks

Online tutorials:

- Basic JavaScript: http://www.2ality.com/2013/06/basic-javascript.html
- Code.org: http://www.code.org/
- AppendTo: http://learn.appendto.com/
- LearnStreet: http://www.learnstreet.com/
- TreeHouse: http://teamtreehouse.com/ (We have 50% discount for them, http://trhou.se/YUL5Xs or http://trhou.se/TcMODa)

Books:

- Eloquent JavaScript: http://eloquentjavascript.net/
- JS, the Good Parts: http://www.amazon.com/JavaScript-Good-Parts-Douglas-Crockford/dp/0596517742

Hope to see you at future meetups!

Thanks for attending the workshop! The slides are available at:
http://tinyurl.com/gdisf-javascript-intro
We will try to always keep them available at that URL.

We'd love your feedback on the workshop. If you have a few minutes, please provide it here (or just reply to this email):
https://docs.google.com/spreadsheet/viewform?formkey=dG5JSlZfaVhNaTlSVmU0S0lHd3NiZUE6MQ

To continue your learning, here are some recommended resources:

These workshops are designed to follow Intro to Javascript, in order so if you'd like to continue your journey:

- Advanced JS: Intro to jQuery

- Advanced JS: AJAX Workshop
- Advanced JS: Working with APIs
- Advanced JS: Intro to MVC Frameworks

Online tutorials:

- Basic JavaScript
- Code.org
- AppendTo
- LearnStreet
- TreeHouse: (We have 50% discount for them, http://trhou.se/YUL5Xs or http://trhou.se/TcMODa)

Books:

- Eloquent JavaScript
- JS, the Good Parts:

Hope to see you at future meetups!

Follow-up Email after JS 200

The page with all the links is available at:
https://dl.dropboxusercontent.com/u/10998095/js200-oct_27_2013/index.html
We will try to always keep them available at that URL.

Other links we used:
http://www.nievie.com/mad_libs/ (Mad libs)
http://jsbin.com/OXeQUBo/1/edit (Kittens!)
http://jsbin.com/eLubUwO/1/edit (More kittens)

To continue practicing your JS, check out Bianca's weekly office hours at HackReactor:
http://www.meetup.com/Women-Who-Code-SF/events/139640372/

We have many advanced JS workshops coming up in the next few weeks, starting with tomorrow night's workshop on jQuery. These workshops go in sequence, building on what we covered on Sunday. We'll likely re-offer them again in February, if you want to spend more time absorbing what you've learnt so far. See the full list here: http://www.meetup.com/Girl-Develop-It-San-Francisco/

Besides our workshops, there are tons of ways to keep learning JavaScript online:

Online tutorials:

- Lynda
- AppendTo
- LearnStreet
- TreeHouse (We have 50% discount for them, http://trhou.se/YUL5Xs or http://trhou.se/TcMODa)

Books:

- Eloquent JavaScript
- JS, the Good Parts:

Hope to see you at future meetups!

Follow-up Email after jQuery class

Thanks for attending the workshop! The slides are available at:
http://teaching-materials.org/jquery/
We will try to always keep them available at that URL.

We both reviewed and covered a lot last night, and for those of you who learnt JavaScript just this weekend, you're probably feeling overwhelmed by the amount of information in your head. Just know that it's really impressive how much knowledge you've exposed yourself, and don't worry if it all seems a little crazy. It is! :-)

I recommend doing "Exercise 3" as homework, and if you have time, continue learning and practicing jQuery from online resources listed below.

Online resources:
http://jqfundamentals.com/
http://www.codecademy.com/tracks/jquery
http://net.tutsplus.com/tutorials/javascript-ajax/15-resources-to-get-you-started-with-jquery-from-scratch/
http://learn.appendto.com/lesson/selectors-101
https://cooperpress.com/

To continue your learning, you can come to the upcoming series of workshops over the next few weeks, all

listed at: http://www.meetup.com/Girl-Develop-It-San-Francisco/

Hope to see you at future meetups!

Follow-up email after AJAX:

Thanks for attending the workshop! The slides are available at:
http://teaching-materials.org/ajax/
We will try to always keep them available at that URL.

This Thursday's JS APIs workshop will show how to do cross-domain requests and get data from other servers: http://www.meetup.com/Girl-Develop-It-San-Francisco/events/143861392/

Here are some relevant online tutorials on AJAX:
http://www.lynda.com/AJAX-training-tutorials/152-0.html
http://www.jquery4u.com/json/ajaxjquery-getjson-simple/

Hope to see you at future meetups!

Follow-up email after JS APIs:

Thanks for attending the workshop! The slides are available at:
http://gdisf-js-apis.appspot.com/
We will try to always keep them available at that URL.

If you try out the final exercise and have any questions on it, feel free to put the code up on github, jsbin, or jsfiddle, and email me with questions. I encourage you to look around your blog, website, company's site, etc, to find ways you can sprinkle APIs (and decide if it's worth it, of course).

You can also do codelabs on Codecademy that use APIs - search for the ones on this page that list "JavaScript":
http://www.codecademy.com/tracks/apis

Hope to see you at future meetups!

Follow-up Email after Introduction to Designing User Experiences

Hey everyone, Thanks for attending the workshop!

The slides from Saturday are available at: http://bit.ly/introux-slides

To continue your learning, here are some recommended resources.You can view our full list at: http://bit.ly/introux-resources

Websites & Blogs

- Smashing Magazine - http://www.smashingmagazine.com/
- A List Apart - http://www.alistapart.com/

Books

- Don’t Make Me Think (book) - http://www.amazon.com/Dont-Make-Me-Think-Usability/dp/0321344758
- The Inmates are Running the Asylum - http://www.amazon.com/The-Inmates-Are-Running-Asylum/dp/0672316498

Let us know if you have any questions. Hope to see you at future meetups!

- Gwen & Sheba

Thanks for attending the workshop! The slides are available at: http://www.teaching-materials.org/jsmvc/

If you have questions about your exercise, stick up the code on a URL (like using jsbin.com) and shoot me an email.

To continue your learning, here are some recommended resources: (let me know which ones you find more useful than others, by the way!)

On OO javascript:

Eloquent JavaScript: Object-Oriented Programming Chapter
MDN: Working with objects
MDN: Inheritance and the prototype chain

On “this”:
MDN: this
JS “this” in different contexts

On MVC:
Journey through the JavaScript MVC Jungle
TodoMVC

On Backbone:
backbonetutorials.com
HelloBackbone Tutorial
BackboneJS Slides
Backbone Wine Cellar Tutorial
BackboneRails (Episodes 1 and 2, in particular)

On Knockout, a MVVM framework: LearnKnockout
On AngularJS: http://www.egghead.io/

Hope to see you at this Thursday’s Backbone meetup or other GDI workshops in the future!
- pamela

Follow-up Email for JS Backbone Workshop

Thanks for attending the workshop! The slides are available at:
http://www.teaching-materials.org/backbone/

I encourage you to keep going with the exercises. If you have questions about them, stick up the code on a URL (like using jsbin.com) and shoot me an email. There will also be a PairUp on August 3rd that you could practice Backbone at.

To continue your learning, here are some recommended resources:

On MVC:
Journey through the JavaScript MVC Jungle
TodoMVC

On Backbone:
backbonetutorials.com
HelloBackbone Tutorial
BackboneJS Slides
Backbone Wine Cellar Tutorial
BackboneRails (Episodes 1 and 2, in particular)

On REST APIs:
Wikipedia: RESTful Web Services

You might also be interested in the slides or the video of a talk on how we use Backbone at Coursera.

Hope to see you at future meetups!
- pamela

Thanks for attending the algorithms workshop! You can reach the slides at
http://www.teaching-materials.org/algorithms/

Here are a few links that were mentioned in class:
http://similarbooks.appspot.com/
http://theinfosphere.org/Futurama_theorem#Solution_algorithm_in_plain_English
http://www.cleveralgorithms.com/
http://xkcd.com/1185/
http://xkcd.com/1185/
http://www.cs.cornell.edu/home/kleinber/networks-book/
http://www.oberlin.edu/math/faculty/bosch/making-tspart-page.html

And here are some other follow-up materials:

- Coursera Algorithms classes (I like the Algorithms Part 1 from Princeton)

- JavaScript and Computer Science
- WomenWhoCode Algorithms Study Night
- "Cracking the Coding Interview" (and similar books)

I recommend trying to use a few algorithms yourself - like by putting together a visualization using JS or using them in a project you're already working on or dreaming up.
Happy algorithming!

Thanks for attending the workshop! The slides are available at:
http://www.teaching-materials.org/jsoo/#/

Here are some follow-up resources on Object-Oriented Programming in JS:

- Eloquent JavaScript-"Object Oriented Programming"
- MDN: Intro to Object-Oriented Programming

To practice, try creating your own objects. Look around your house and object-ify it!

The MVC workshop on Wednesday is a great follow-up for this one, you can RSVP for that here:
http://www.meetup.com/Girl-Develop-It-San-Francisco/events/143863352/

Hope to see you at future meetups!

Thanks for attending the workshop! The slides are available at: http://bit.ly/GDIjstesting

We will try to always keep them available at that URL. We'd love your feedback on the workshop. If you have a few minutes, please provide it here (or just reply to this email):

https://docs.google.com/a/fionatay.com/spreadsheet/viewform?formkey=dHNZc0w1WnhRRklRQW1jckxsOHFEVEE6MA#gid=0

To continue your learning, here are some recommended resources:

http://pivotal.github.io/jasmine/

http://evanhahn.com/how-do-i-jasmine/

http://visionmedia.github.io/mocha/

Follow-up email for CSS Tools/Techniques

Thank you for attending the workshop!The slides are available at:
http://www.teaching-materials.org/htmlcss-1day/lesson6/slides.html#slide1
I've updated them with links to additional resources that we discussed, particularly on data URIs and custom fonts. (If you're familiar with Git, you can view the diff on the slides here: https://github.com/pamelafox/teaching-materials/commit/b0cf5a7b47315423146d745f6dc9408999e1bbf4)

If any of you have existing website or blogs, think about how you can apply these tools to your blogs. Maybe check how they look in print mode or on your phone and see how you can improve them, and try using a framework like Bootstrap as the base CSS for your site.

For those of you who want to continue practicing CSS, Codecademy has some nice exercises: http://www.codecademy.com/tracks/web and Treehouse has great videos: http://teamtreehouse.com/

You could also sign up for our next PairUp, which is a fun excuse to practice what you've learnt: http://www.meetup.com/Girl-Develop-It-San-Francisco/events/130475592/

See you at future workshops!

- pamela

Thank you all for attending the CSS3 workshop! The slides are available at:
http://tinyurl.com/gdisf-css3

Here are some resources for further reading:
On relational selectors

- Good explanation of child, descendant, and sibling selectors

- Another good explanation of advanced relational selectors

On Attribute Selectors

- Good overview of attribute selectors with examples of how you might use them

Structural Selectors

- Nth Master (gives examples of all sorts of formulas you can use to do amazing things with nth-child)
- Explanation of the difference between Nth-of-Type and Nth-Child

Other helpful CSS3 resources

- HSL Color Picker
- CSS3 Please (enter the values you want for a property once and it gives you the prefixed code to use them across different browsers)
- CSS3 Generator (a good tool for experimenting with properties like border radius, shadows, and gradients)

Keep practicing!