Prepare to learn

Web Development Boot Camp

Lesson 1.1

How to use our Slack channels

- Slack is our main form of communication
  - `#01-general` is for general announcements
  - `#02-ask-the-class` is for asking and answering your peers' questions
  - `#03-resources` is for sharing tools and further reading
  - `#04-shout-outs` is for recognizing
  - `#05-questions-realtime` - ask for answer during lecture, and upvote
  - `#06-class-repo` - notifications of new content added to class repo
  - `#07-study-groups` is for you all to organize study groups
  - `#08-random` is for fun
  - `#09-office-hours-signup` is for getting help in office hours
- Use threading when appropriate (cuts down on noise)

Announcements - Zoom

- When you have a question, use the reactions on the bottom of the Zoom window to raise your hand

- There is no Zoom chat: use Slack

- Please have your cameras on

Announcements - John’s Schedule

John on Vacation

June 20th - June 27th

(Week 3: Javascript)

Sean and I are working on a replacement for that week

Why are you here in this class?

Check for a link in the `#01-general` channel

This is a problem solving class.

We use code and technology to create solutions.

Example: Bread

What problem does bread solve? **Hunger**

How do we create the bread?

Ingredients: Flour, Water, Salt

Tools: bowl, measuring cup, spoon, oven

Recipe: a set of instructions

Where did those instructions come from?

Experimentation, trial and error, optimization, documentation

Do we have to do that everytime we make bread?

No - we reuse that solution and extend it

Example: Web Page

What problem does a web page solve? **Organize and present information**

How do we create a web page?

Ingredients: HTML, CSS, JavaScript

Tools: computer, web browser, text editor

Recipe: a set of instructions to give to a browser in a computer

Where do the instructions come from in web development?

Experimentation, trial and error, optimization, documentation

Do we have to do that every time we make a web page?

Yeah actually. We can reuse patterns, algorithms, frameworks but still need a custom solution

What code can I reuse?

Acceptable code reuse

Common forms of reusable code we will use in this class

- Examples from documentation (e.g., Mozilla Developer Network)
- Blog articles that explain or demonstrate (e.g., how to configure a server, use a library)
- API’s (application programming interface)
- Frameworks (e.g., Bootstrap)
- Stack Overflow

Copying someone else's solution is not-acceptable

That is plagiarism and clearly won't help you in the long run.

Unacceptable code reuse

Forms of plagiarism

- Copying code from someone in the class (if you work together, please call it out in comments)
- Finding a previous Bootcamp's solution, copying it and slightly modifying it
- Finding a tutorial on line
- **Anything that avoids developing your own solution**

What about AI?

A.I. can solve simple problems but using it at this point in your career takes away your opportunity to learn coding as a skill

Is this class a good investment?

Languages and technologies come and go but problem solving doesn't go away

What should you expect?

Do not expect to get a software job right out of this class

There is no job guarantee

- You are a developer at the beginning of a journey
- Completing the bootcamp accelerates your progress but it is not the end
- You will need to learn how to interview
- You will likely need to learn more skills

Expect to work 12 to 20 hours a week outside of class

I know. It is a real challenge.

There is no “slide by” in this class

This isn’t high school or even college for that matter

- Doing homework in class won’t work
- Plagiarism is not allowed and it is easily discovered
- This is skill-based. Your product either works or it doesn’t
- Project teams are matched based on class engagement
- If you think this will be easy...it won’t be

This s**t is hard

There is good news

Doing hard things is good for you

- It builds confidence
- You gain mental toughness
- You find your potential
- You learn how to deal with pressure

You will learn a lot on your own

We can't teach everything but we can teach you enough to learn on your own

You are now a

Professional Learner!

This Should Be You

Learn to Learn

Boost these four attributes

- **Aspiration**
- **Self-Awareness**
- **Curiosity**
- **Vulnerability**

You are allowed to not know the answer.

(you are paying for that privilege)

You are allowed to be curious

You are allowed to ask questions

You are allowed to tinker.
Experiment.
Play with these new toys.

The Great Confusion

My code doesn't work and I don't know why.

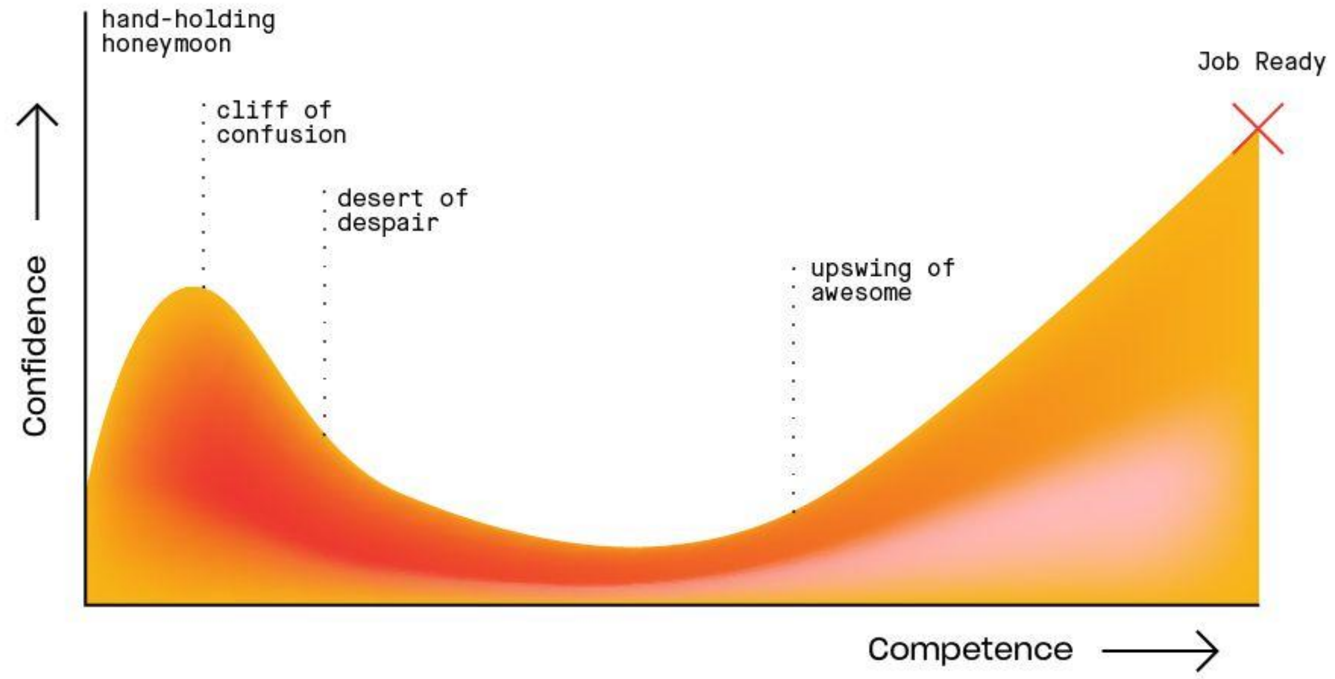
My code works and I don't know why.

The Great Doubt: Imposter Syndrome

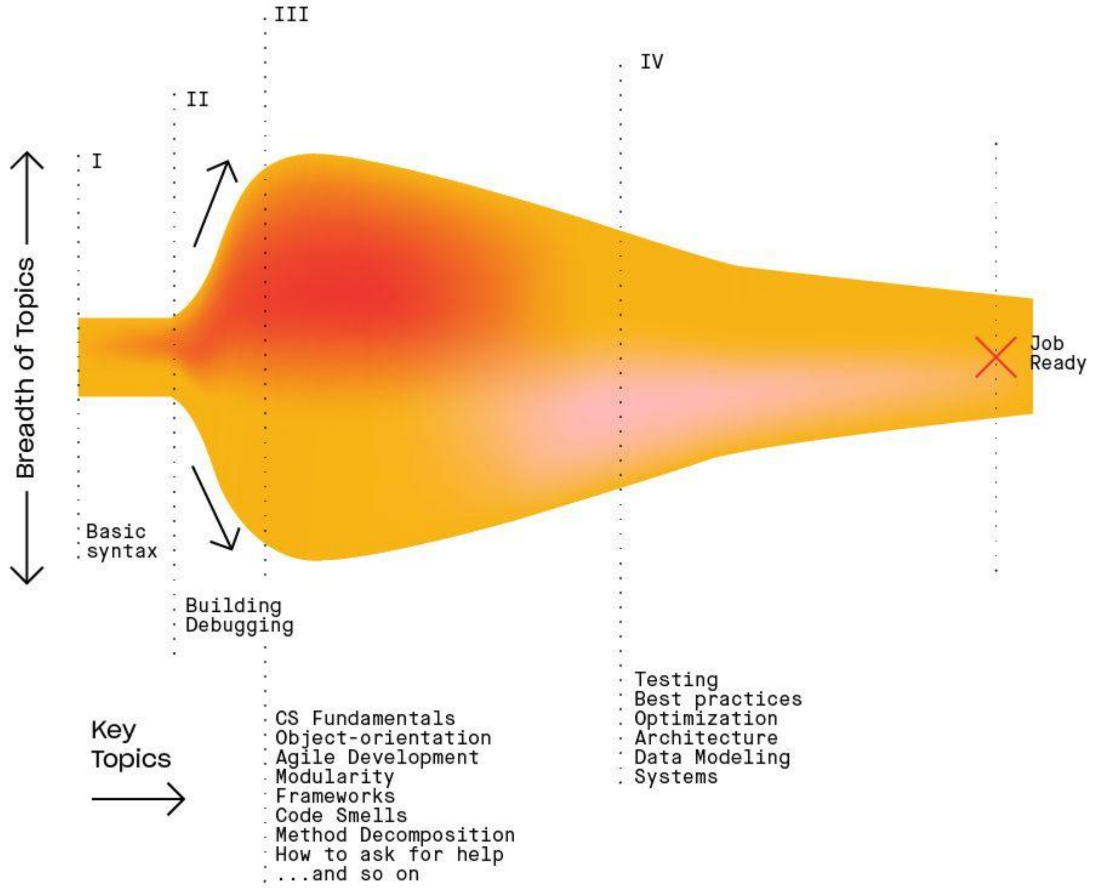
“Maybe I’m just dumb.”

Obstacle #3: The Great Distance

Coding Confidenct vs Competence



Scope of Knowledge



CS Fundamentals
Object-orientation
Agile Development
Modularity
Frameworks
Code Smells
Method Decomposition
How to ask for help
...and so on

Testing
Best practices
Optimization
Architecture
Data Modeling
Systems

Full-Stack Development

The Browser	Dev Tools	Server Side
HTML	Heroku	Templating engines
CSS	Git	Sessions
JavaScript	GitHub	Writing tests
jQuery	**Databases**	Node.js
Bootstrap	MySQL	Express.js
SEO	MongoDB	Creating APIs
API Interaction		MVC
APIs (Consuming)		User authentication
JSON		ORM (Object-relational mapping)
AJAX		**CS Fundamentals**
Cutting-Edge Development		Design patterns, Algorithms
Progressive Web Applications		
React.js		

Nothing Comes Easy

Learning to code requires two things:

Persisting in the face of something that feels incredibly hard and confusing.

Maintaining the self-confidence necessary to believe that **YOU CAN DO THIS.**

You are going to fail. A lot.

A master has failed more times than a beginner has tried...

Get used to feeling frustrated

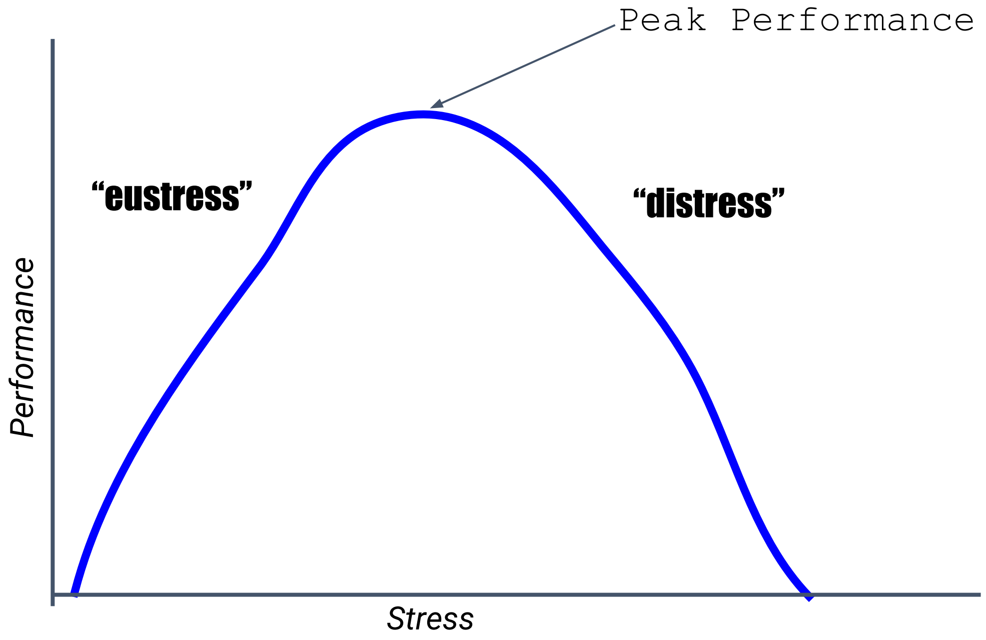
You can't tell whether you're learning something when you're learning it—in fact, *learning feels a lot more like frustration*.

What I've learned is that during this period of frustration is actually when people improve the most, and their improvements are usually obvious to an outsider. If you feel frustrated while trying to understand new concepts, try to remember that it might not feel like it, but you're probably rapidly expanding your knowledge.

—Jeff Dickey, author of *Write Modern Web Apps with the MEAN Stack: Mongo, Express, AngularJS, and Node.JS* (Peachpit Press, 2014)

Stress: Eustress vs Distress

Some stress is needed, but too much and we crack



How much struggle is enough?

You need to struggle, to find answers yourself, but too much struggle will lead to distress

- Spend at least an hour, then please reach out for help.
- Everything you find that doesn’t work helps build your intuition
- Not finding the answer is also expanding your knowledge and vocabulary
- Allow yourself to do this in spite of deadlines, etc.

Google Fu:

What Is Google Fu?

AI Prompts:

phind.com

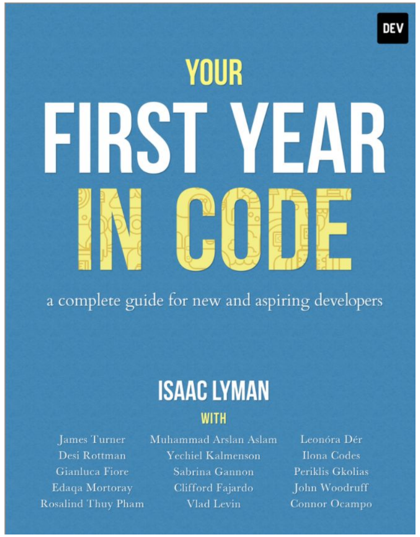
And Remember...

If you want to go fast, go alone. If you want to go far, go with a team.

Great advice

Your first year in code - Issac Lyman

Read Chapter 3 - it is in the class repo



- Build Intuition
- Knowledge debt
- Problem driven learning
- Multiple sources of truth
- Fewer subjects at a time
- Ask your own questions
- Challenge the material
- Go back and review
- Fundamentals are important

What do we expect of you?

We expect you to try

Trying looks different for everyone, so try at your own level

I WANNA CODE BUT I DON'T KNOW HOW TO GET STARTED!
LTURD.COM

We expect you to show up

Missing a class here and there is ok. Longer absences are harder to recover from

Let us know what’s going on.

Open communication is the best strategy

Cameras should be on at all times

This is how we know you are here and that you are participating in the activities

If that isn’t possible, please let us know

Open communication is the best strategy

Support Team Promise

If you're willing to put in the time and take our advice, we're here to help you 100% of the way.

This goes for everyone working on this program:

- Instructors
- TAs
- Student Success Team
- Career coaches
- Everyone else!

Advice from former students

Feedback from former students

"Learning how to use Google, like, REALLY use Google, is just as important as learning JavaScript or Express or any other technology we learned during boot camp. It's essential!"

"I would tell myself to re-do all the class activities I got stuck on. Do research going into the week on the topics you will be covering that week. And make sure you can answer all the questions in the weekly unit README at the end of the week to make sure you grasped the concepts from that week. I would also tell myself that it's okay if you have no experience coming in. If you work hard and use your awesome instructional team and all the other resources the bootcamp has to offer you CAN do it!"

Feedback from former students

“ This is not going to be easy at all. But when you feel like you can't do it anymore, KEEP GOING! When you need help, ask for it, stop being so damn prideful. Things that seem easy for everyone else, are going to be hard for you; however, things that seem hard for everyone else will be easy for you. You're going to be fine. Trust what John, Ben and Abdul say. Ask for help, I can’t emphasize this enough. You're going to finish the class and from there, I don't know, but you're about 1,000 steps closer to your goals than you were before.”

Feedback from former students

“This class is a foundation. Don't expect to know any technology fully. The class is a launching point, not a destination. Trust the process, it takes time and requires your constant effort, but it works.”

“it's hard and you know it will be, but it's actually harder than you can imagine. you can get a 2nd tutor, request a change if you don't like your first one!!! I didn't know and i would def been further ahead!!!”

“Ask for help when you need it, don’t wait until the last minute because the material goes by very fast! Also breaks are very important, and you got this!”

Feedback from former students

1. This course is very intense/demanding for some and kind off medium difficult for others. As long as you don’t compare yourself and just try to just compete with yourself you will be okay (Something my dad taught me, and I religiously believe in). I think i personally found this bootcamp to be very demanding and if you cant give 4+ hours daily for atleast 5 days a week you can’t succeed.
2. Redoing class activities is THE MOST USEFUL THING to keep up with class pace. Practice makes it easier
3. It is okay to not understand a word of some concepts sometimes(Express and REACT hooks) . I panicked and you know that lol but it took me several weeks and hours of bugs to understand just ROUTES !Those classes have now taught me that the stuff we learnt was hard, everybody is different.
4. Next best help is TUTOR and OFFICE HOURS . Try to be there and ask as many questions as you can. Don’t care who thinks what of you.
5. Reading DOCUMENTATION of any new topic introduced helps cuz your instructor/TA will not always be there for you.You need to learn by yourself .
6. Completing homework on time helped me panic less and keep up with class pace. I didn’t have to worry about pending homeworks during projects and stuff.

Hope this helps and all the very best for your new venture !

Feedback from former students

“The more time you put in coding, the easier it becomes. You can’t just read documentation and listen in class and get it. You may understand the concepts, but to become a better coder, it just takes time and effort. Anything that is “hard” just means you need to dedicate more time to developing the skill.”

Feedback from former students

DO YOUR HOMEWORK AND TURN THEM IN ON TIME

Make friends, persevere, you're not a bad coder you just need a second pair of eyes to solve a simple problem sometimes!

Keep up on HW. Don't be afraid. Be confident. And Dive In!

Don't cry over JavaScript

Yeah, keep on top of HW!!

Coding is an emotional rollercoaster, one minute you will feel like you're on top of the world and the next like you're an idiot and don't know anything. HAVE GOOD STRESS MANAGMENT

Appreciate how cool what you are doing is!

Try to get your hands dirty all the time! Do not stop coding!

Test often and look at every line and EVERY UNDERSCORE AND DASH, would have saved me hours upon hours

if you get stuck. pseudo code or write out step by step what you are trying to do

Commit after every feature/layout change! Even if you don't push it until later, VSCode will have the timeline so you can revert back to something that worked but got taken out along the way.

Feedback from former students

You will feel like you don't know what you're doing, you will feel like you are behind and that you have no idea what's going on, and the TRUTH of this matter is simply that it is NOT the case! My brain seems to focus on what I do not know instead of all the things that I do know, and it makes it seem far scarier than it needs to. I have now graduated and can say with confidence that I can make websites - I STILL feel like I know very little! Keep going!!!

Try to watch a crash code video on the topic before class

Practice coding outside the homeworks, too! Grasshopper, HackerRank, anything to keep yourself thinking of different ways to solve problems

If you can't figure out a bug in an hour, STOP STARING AT IT. Go to bed. It will be so much easier coming back to it with fresh eyes

It may seem hard at first.. but you will eventually get it!

@6 month ago me: go through each line of code and make sure you understand what's happening

You will have imposter syndrome. You will feel inadequate. Just start your homework. That's all I can say. Just start your homework even if you don't know how. Trust the process and with time you will develop the knowledge necessary. Also, turning in a homework on time doesn't mean anything if you don't understand it. Turn it in late if you need to, just understand it.

hang in there, research whatever you don’t understand, don’t be afraid to try, and push through it.

Recent Grad Activity

- Recent grad (a journalist) is now a Technical Writer
- DevOps at Wizards of the Coast
- Dev at zulily
- Interview project - when she came to this class she was an accountant, now has a front-end AngularJS job (which we don't teach in this class)

  1. I created a light back-end with Node+Express to FS read the file and used Xml2JS parser to convert to an object.
  2. I used Angular for the front-end to make an HTTP request every minute to the pathway that I sent the xml object and display to user.
  3. The challenge was making sure that the components showed the most up to date info when the request was made. Angular has a concept of using a service in the app to "bind variables" so that the components only pull the updated version of it. I made sure that I made the HTTP request in one place only rather than slow down the app with each component making that. Also, accessing the data required all the practice that I've been doing with Leetcode - some data was in an array of arrays with one index having sub data, or an array of objects with inconsistent naming conventions ($ sign for first key and the data was another layer in)
  4. I also had to address how to handle errors (the format of the xml file changes). I rendered each table dynamically so that if another field was added to the underlying file, it automatically rendered an additional column and related field.
  5. Lastly I made the tables responsive so that on a tablet or mobile phone it adjusts to still be readable.

“ I found that once you have the foundation of code logic, you’re good to go.”

Q. What is a grade?

Course Structure

Daily Schedule

Don’t Forget Office Hours - I usually stay later than I should

01

Pre-Class Office Hours:

- Questions in a queue for help (usually a thread in the general channel)
- We will help in the order they come in

02

Class Time:

- New instruction
- In-class activities

This is not the time to do homework.

Really.

03

Post-Class Office Hours:

- A new queue
- Continuation of pre-class questions

Daily Schedule - In Class

In each class, we'll cover the following:

01 Set objectives

02 Background lecture

03 Watch me/coding demos

04 Code discussions

05 In-class exercises

06 Project work

Daily Schedule

In each class, we'll cover the following:

01 Set objectives

02 Background lecture

03 Watch me/coding demos

04 Code discussions

05 In-class exercises

06 Project work

This is the super important stuff—ALWAYS BE CODING!

Prework

Software Checklist

At this point, you should have all of these installed:

- ☐ Slack
- ☐ Visual Studio Code
- ☐ Git
- ☐ Git Bash (Windows) or Terminal (Mac)
- ☐ Google Chrome

Accounts Checklist

You should also have accounts for:

- ☐ GitHub (with SSH Integration)
- ☐ LinkedIn
- ☐ Stack Overflow

Self-Check

Let’s do some quick checks of the following:

- ☐ Visual Studio Code Check
- ☐ Git Bash/Terminal Check
- ☐ Git Check