Introduction to JavaScript

Programmatic Thinking

Think back to a time when you tried to give the “perfect” gift. Maybe your thought process was something along these lines:

What would they think of this shirt? No, that's too bold to wear to the office. How about these headphones? No, the earbuds would fall out when they exercise.

And so on. To give that perfect gift, you have to think about the person who's getting it.
Similarly, to be an effective programmer, we have to understand how a computer thinks. The good news is that computers are a lot more predictable than people.
So, what do we mean when we say "programmatic thinking"? And what about just "programming"? Let's watch the following pair of videos and find out.

MyGA | General Assembly

MyGA | General Assembly

Programming vs. Programmatic Thinking

Programmatic thinking is the kind of critical thinking that helps you overcome problems you may face as a programmer. It allows you to address issues efficiently, learn on your own, and problem-solve on the fly.

Programming refers to the the writing of computer programs. The amount of decision-making required in programming varies — if you’re working with a pre-existing framework, it can be minimal, whereas if you're starting with a blank slate, it can be large. Think of the difference between baking a cake from a box versus baking one from scratch.

And, just as there is no right or wrong way to bake a cake, there is no right or wrong way to write a program.

Thinking Like a Programmer

Whether you're making that out-of-the-box cake or a multi-tiered custom creation, you'll eventually run into issues that require effective programmatic thinking.

Let's take a look at four tips for thinking like a programmer.

Tip 1: Remember How to Problem-Solve

- Some of the problems you'll encounter as a programmer will have multiple solutions and require creative and critical thinking.
- The path forward won't always be obvious. Sometimes, you'll have to take a step back and analyze the problem before moving forward. Be flexible and learn from mistakes in the process, too.
- Break the problem into as many solvable pieces as possible. Start with the easiest piece and work up to the hardest one.

Tip 2: Remember How to Break Down Your Code

- A computer is only as smart and efficient as the programmer who writes its code. You have to know how to break big ideas down into small steps and prepare the computer to handle failures before they occur.
- Develop strategies for identifying errors early on. The smallest programming error can keep your computer from running correctly. Such errors are usually easy to fix, just hard to spot.

Tip 3: Remember What You Are Trying to Do

- You have to know what you're doing with the code you're writing before you write it.
- Remember, programming isn't just about creating perfect code, it's also about what you're trying to do with the program you are writing.
- **Pseudocode**, which we will cover in the next lesson, is an essential tool for planning the big picture.

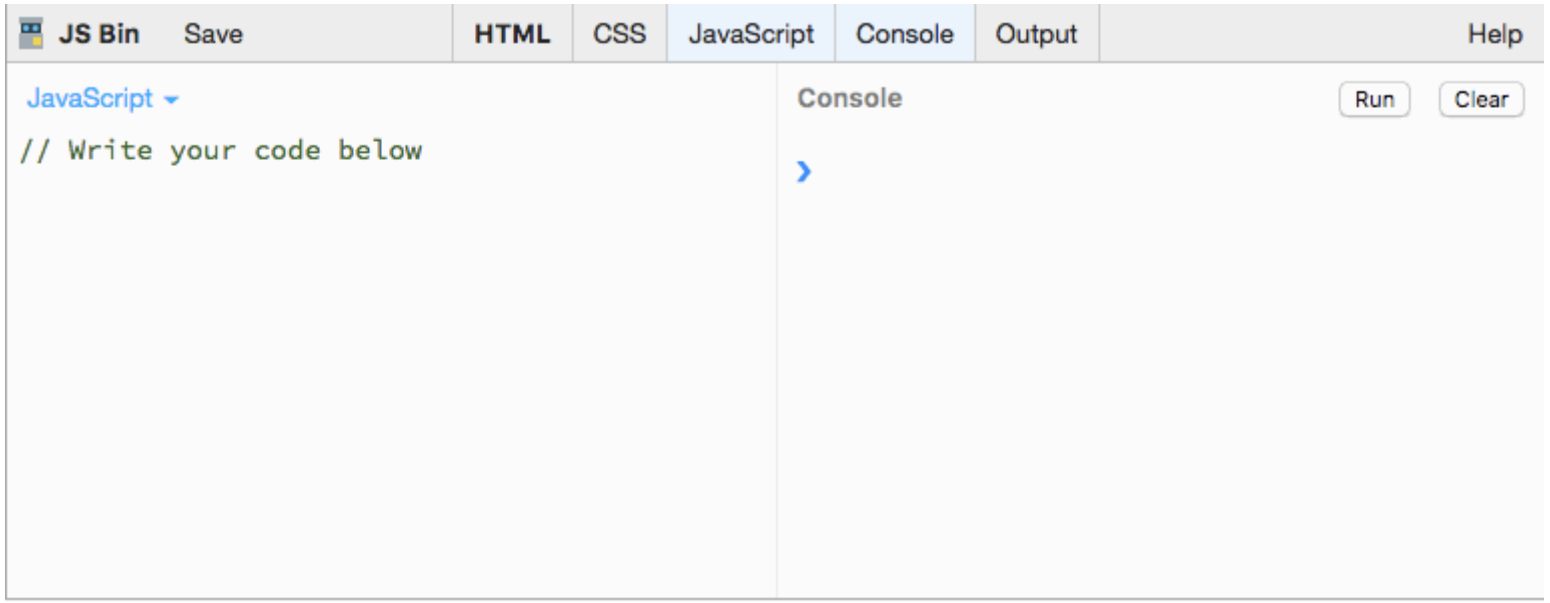
Tip 4: Don't Be Afraid to Use Resources

- A big part of becoming a programmer is learning to be self-reliant, but it's also important to know where to get help when you need it.
- No developer knows how to do everything.
- The world of development is vast and can seem overwhelming when you're a beginner. Focus on the immediate task at hand and the skills that are necessary to solve THAT problem. Learn as you go, and pretty soon new technologies won't feel daunting.

JavaScript

- Throughout this unit, we'll be exploring some of the more common tools and concepts available to programmers.
- Although we'll be specifically looking at these concepts in the context of **JavaScript**, they (or similar ones) are present in nearly every modern programming language.
- Having a strong understanding of the basics is essential and will translate to your future work, whether you end up programming in JavaScript, Ruby, Python, or any other language.

JavaScript in JS Bin



As you can see in the screenshot above, JS Bin features different window panes.
On the left is the editor, or "JavaScript" panel. This is where you'll be writing your JavaScript code.
On the right is the "Console" panel. To execute the code in your editor, click the "Run" button here. This is where you'll see the output of your script.
After writing your JavaScript in the "JavaScript" panel and clicking the "Run" button, you can test values for different variables (which we will cover shortly) by typing the variable name into the right panel, the "Console," and then hitting the "return" key.

Saving with JS Bin

- To save your JS Bin session, click "Login or Register" at the top of the screen, and log in with GitHub. When you select "File," followed by "New," you'll open a blank workspace.
- The moment you begin writing code, JS Bin will generate a URL that you can bookmark and come back to at any time.
- You can hide or display any panels (HTML, CSS, JavaScript, or Console) as needed by clicking on the tabs at the top of the editor.
- https://ga-instruction.s3.amazonaws.com/json/WDI-Fundamentals/assets/unit-7/account.gif

Conclusion

- In this lesson we discussed programming and programmatic thinking (and also a bit about cake baking). We also got acquainted with a new programming sandbox — JS Bin. But, our biggest takeaway is we're now prepared to start coding with JavaScript.
- If you're feeling ready, move on to the next lesson. Otherwise, consider going back and reviewing this material.