GENERAL ASSEMBLY

JAVASCRIPT DEVELOPMENT

Sasha Vodnik, Instructor

JAVASCRIPT DEVELOPMENT

THE COMMAND LINE

LEARNING OBJECTIVES

At the end of this class, you will be able to

- Use the most common commands to navigate and modify files / directories via the terminal window.
- Initialize a local Git repository and push/pull changes to a remote Git repository.
- Run basic JavaScript code on the command line using Node.

AGENDA

Timing	Topic
20 min	JavaScript & Web Technology
20 min	Introduction to the Terminal
20 min	Terminal Codealong
5 min	Break
30 min	Introduction to Git/GitHub
30 min	Git/GitHub Exercise and Codealong
5 min	Break
20 min	Intro to Node and Command Line JS
30 min	Command Line JS Codealong
10 min	Final Questions & Exit Tickets

Homework checkin/questions

- The **most significant thing I learned** in the homework (or last class) is ___________.
- My **biggest outstanding question** from the homework (or last class) is ___________.

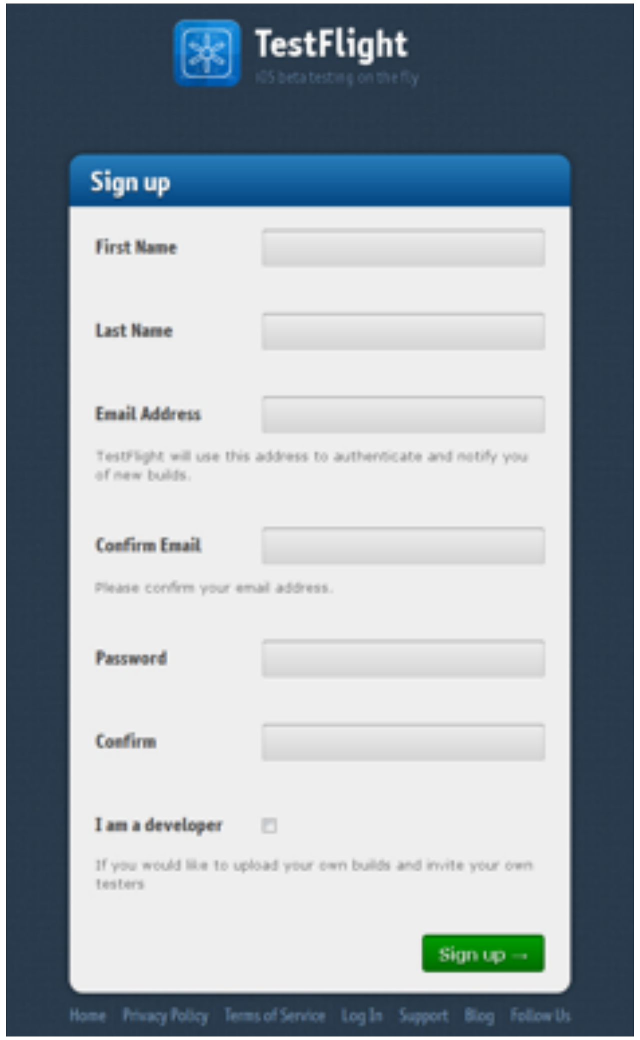
Think about last class:

- We installed software from the command line by typing commands
- We also installed software by downloading an installer, double-clicking it, and following the prompts

Think about the following questions:

- What are some disadvantages to using the command line?
- What are some advantages to using the command line?

WHAT CAN JAVASCRIPT DO?



TestFlight
iOS beta testing on the fly

Sign up

First Name

Last Name

Email Address

TestFlight will use this address to authenticate and notify you of new builds.

Confirm Email

Please confirm your email address.

Password

Confirm

I am a developer ☐

If you would like to upload your own builds and invite your own testers

Sign up →

Home Privacy Policy Terms of Service Log In Support Blog Follow Us

← front end tasks (animations, buttons, forms)

JS

→ APIs, databases, back end tasks

VERY FEW STEPS TO RUN

runs directly in browser within an HTML file

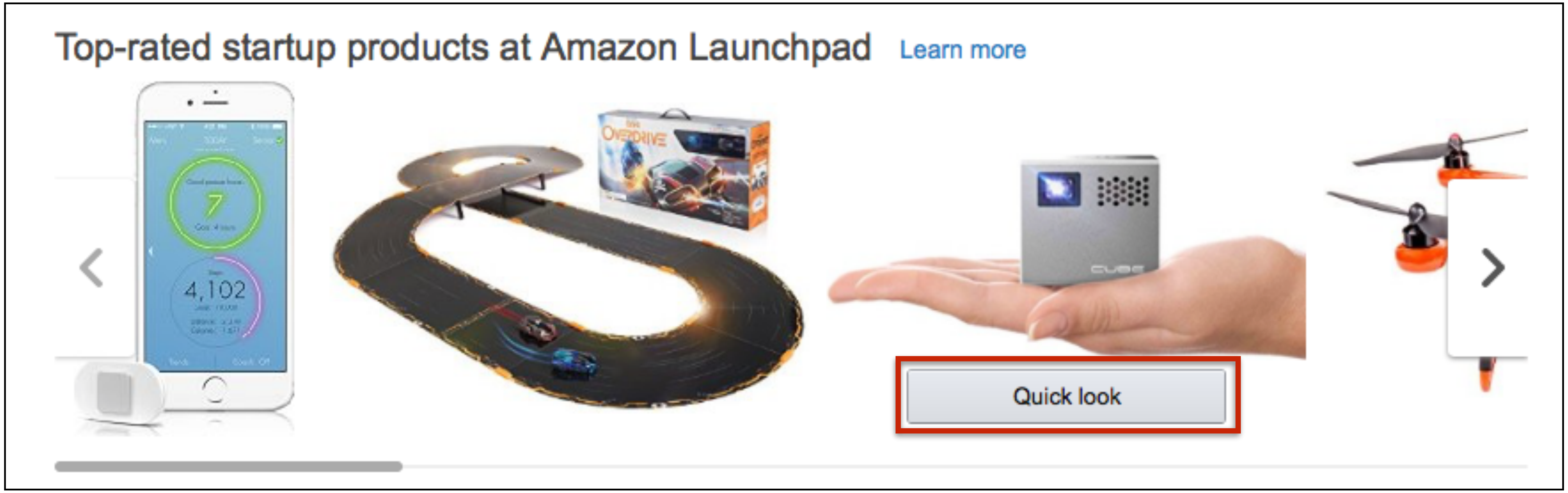
also runs in node.js

AND WORKS EVEN WHEN COMPUTERS ARE OFFLINE

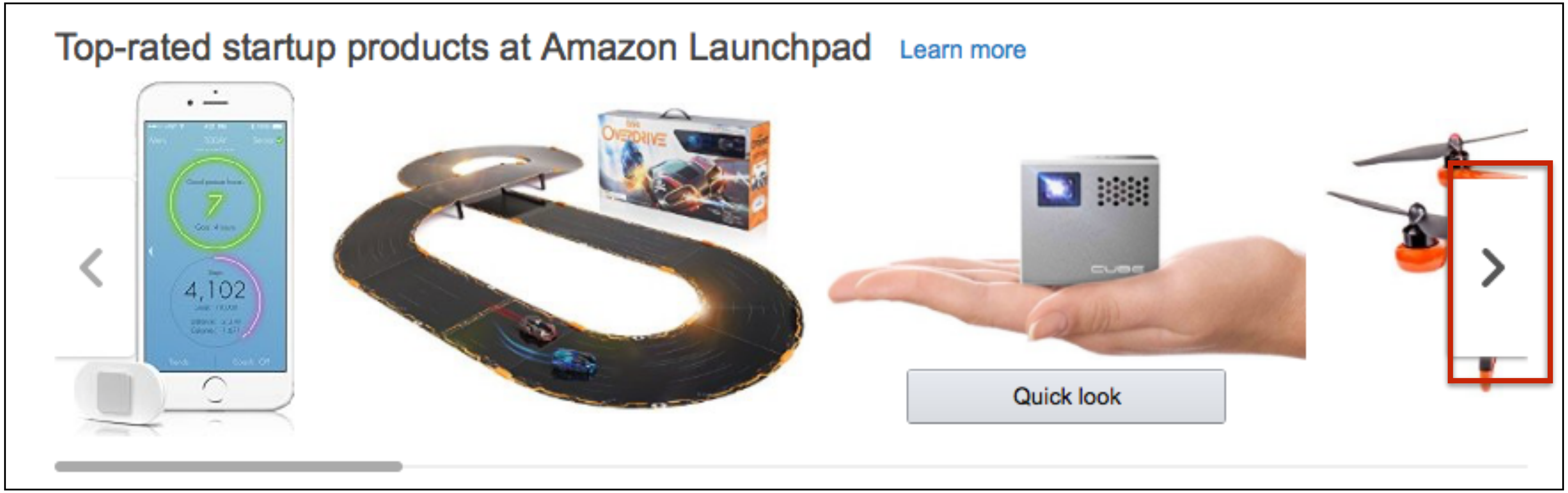
runs directly in browser within an HTML file

also runs in node.js

HIGHLY RESPONSIVE INTERFACES



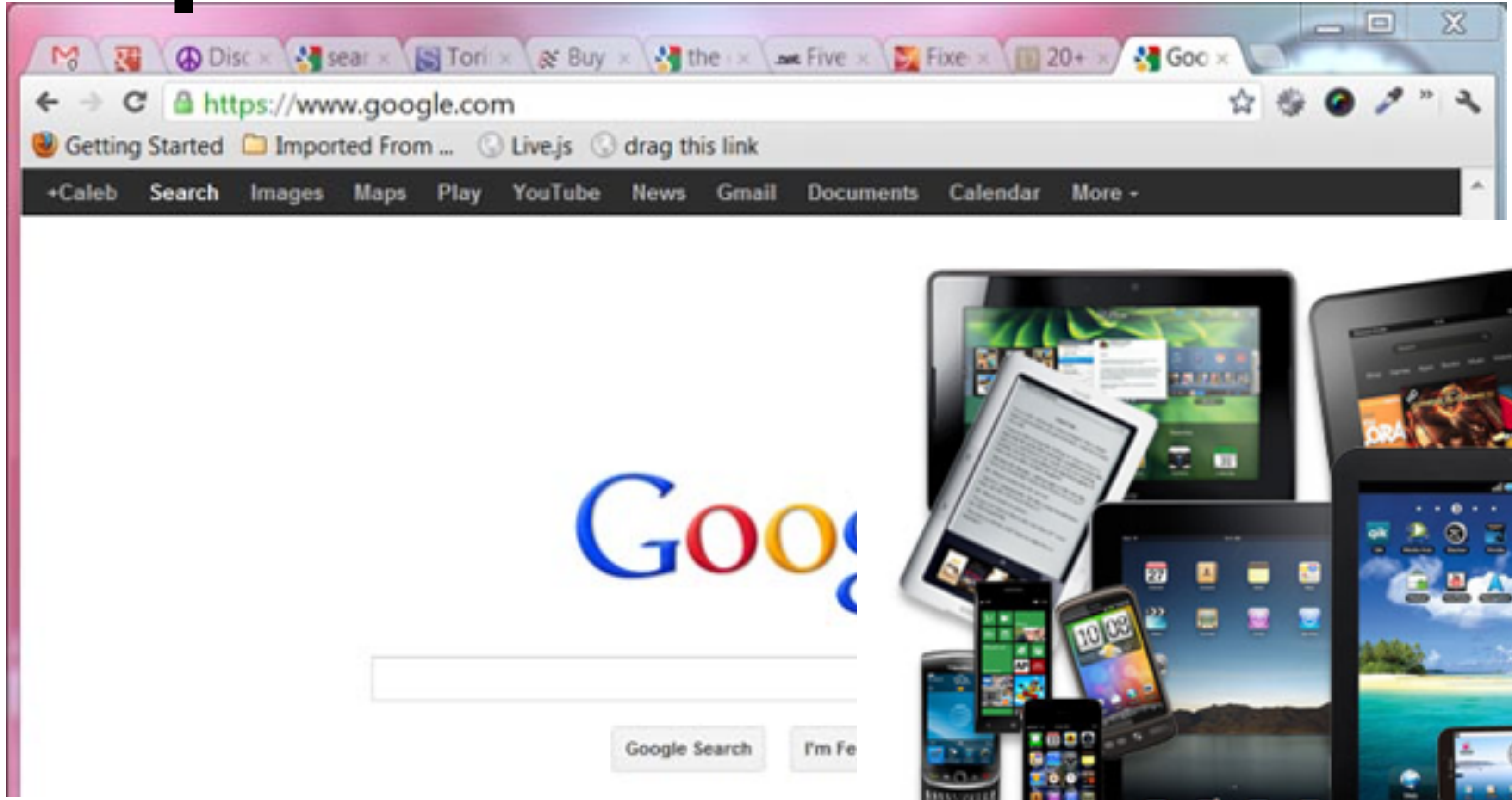
LOAD ADDITIONAL CONTENT WHEN USER NEEDS IT (AJAX)



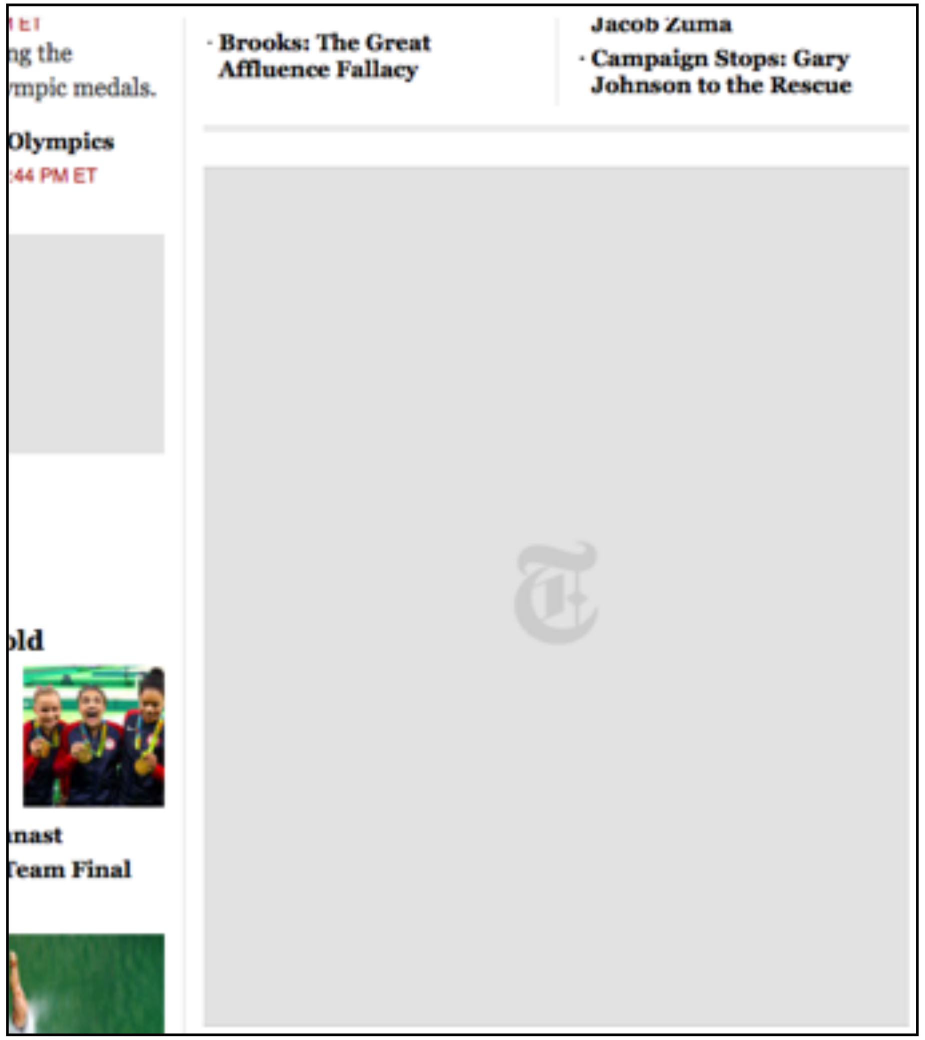
WHAT ELSE CAN JAVASCRIPT DO?

- Determine your browser functional limitations and react accordingly (progressive enhancement)
- Power website backends and physical devices (node.js)

DRAWBACK: The environment in which JavaScript operates is unknown



DRAWBACK: JavaScript can be disabled


ng the
mpic medals.
Olympics
44 PM ET
ld
nast
Team Final
· Brooks: The Great Affluence Fallacy
Jacob Zuma
· Campaign Stops: Gary Johnson to the Rescue

Node.js

Node.js

- An example of where you’ve seen/experienced something you just saw
- A definition (from Wikipedia):
  - In software development, Node.js is an open-source, cross-platform runtime environment for developing server-side Web applications.
- Enables JavaScript on the server (the backend)
- Written in C, C++, and JS (so, not a JS framework)
- Interprets JS using Chrome's V8 engine
- Module driven; see Node Package Manager (npm)
- All about non-blocking, asychronous input/output

Node.js

- We will not be using Node.js as a web server (backend) - see Firebase
- We will be taking advantage of Node's command line interface
- Allows us to run JavaScript from our terminal applications
- More at the end of class...

Node.js

- We will not be using Node.js as a web server (backend) - see Firebase
- We will be taking advantage of Node's command line interface
- Allows us to run JavaScript from our terminal applications
- More at the end of class…

JavaScript Frameworks and Libraries

A Library

- Set of predefined functions that your code calls
- Each call performs work and returns a result (and control) to your code
- Specific, well-defined operations
- Example: jQuery

A Framework

- Opinionated architecture for building software
- Control-flow exists, you fill in with your code
- Calls your code; is always in control
- Examples: Angular and Ember

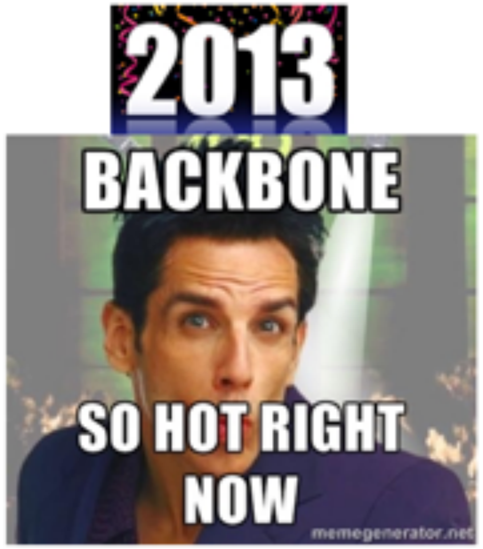
Libraries vs Frameworks

- The primary difference (source):
  - You call library
  - Framework calls you
- Please Note:
  - JSD focuses on the foundations of JavaScript as a programming language
  - We will be using the jQuery library
  - Opportunity towards class end for a framework intro

2007
GWT
SO HOT RIGHT NOW
memegenerator.net

2009
EXTJS
SO HOT RIGHT NOW
memegenerator.net

2012
KNOCKOUT
SO HOT RIGHT NOW
memegenerator.net


2013
BACKBONE
SO HOT RIGHT NOW
memegenerator.net

2014
ANGULAR
SO HOT RIGHT NOW
memegenerator.net

2015
REACT
SO HOT RIGHT NOW
memegenerator.net

Share with a partner

- An example of where you’ve seen/experienced something you just saw
- One thing about JavaScript that is new to you

The Terminal

INTRODUCTION TO THE TERMINAL

- Terminal allows you to interact with your computer faster
- Terminal === Command Line === Console

UNIX

- Family of operating systems, including all Linux systems and OS X/ macOS

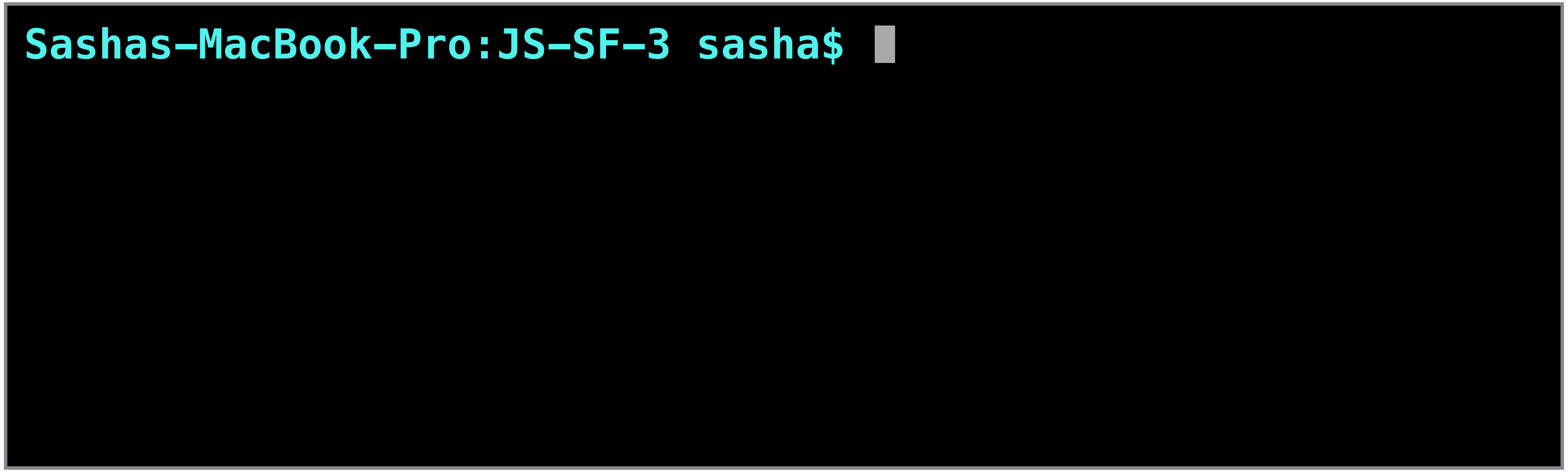
SHELL

- A generic name for the primary program that runs inside a terminal

BASH

- Bourne-again shell: a specific shell program

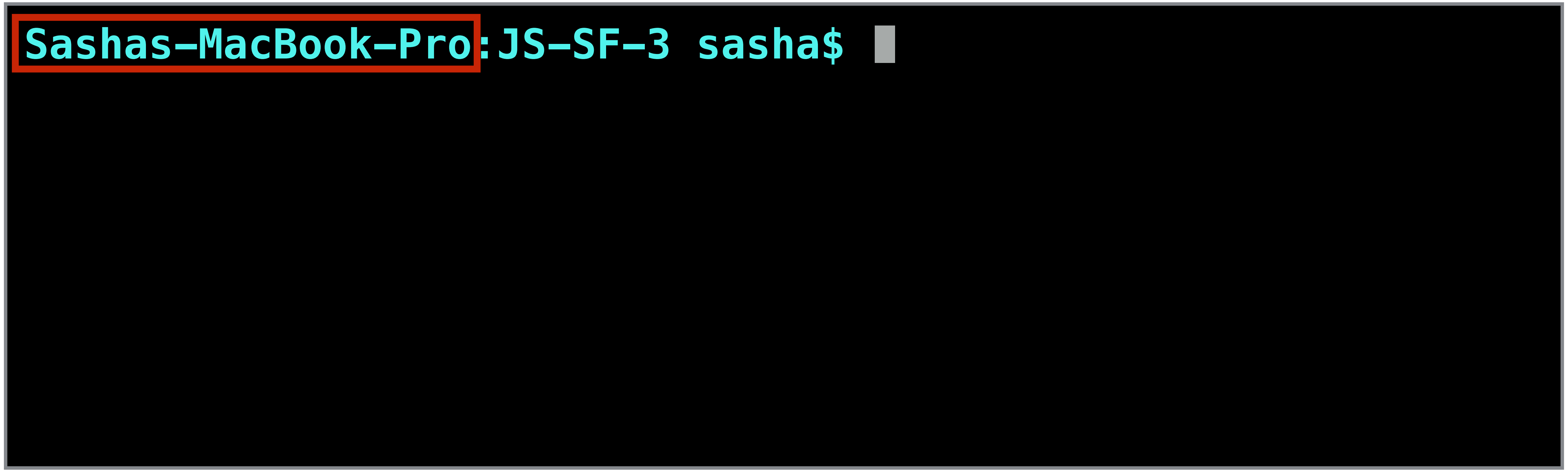
ANATOMY OF THE TERMINAL



```
Sashas-MacBook-Pro:JS-SF-3 sasha$
```

ANATOMY OF THE TERMINAL

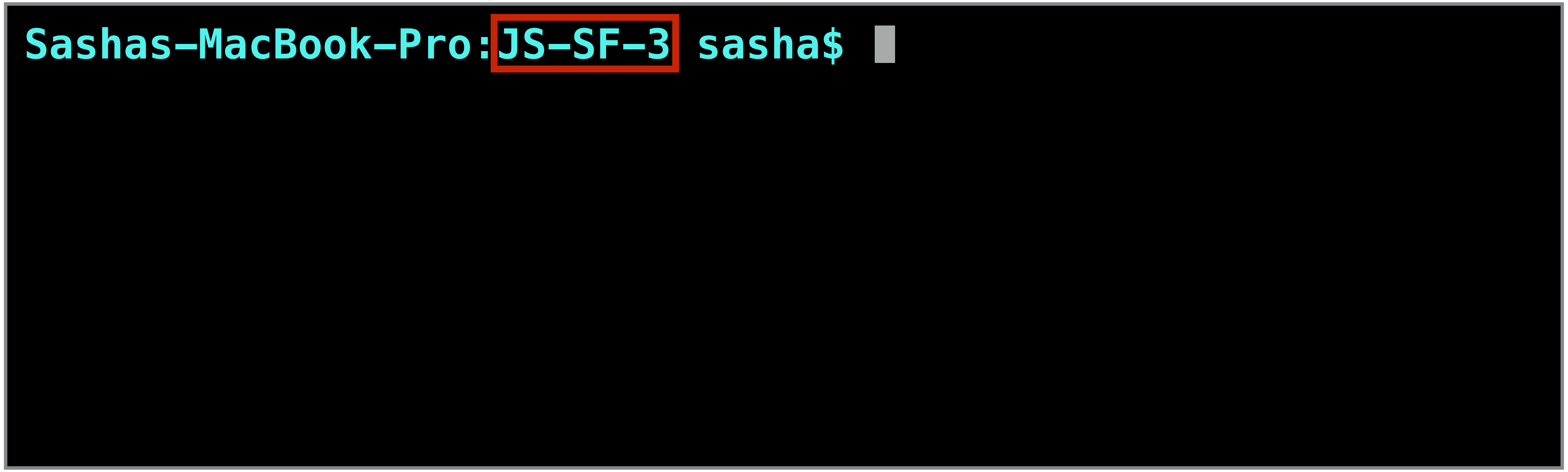
Host (computer) name



```
Sashas-MacBook-Pro:JS-SF-3 sasha$
```

ANATOMY OF THE TERMINAL

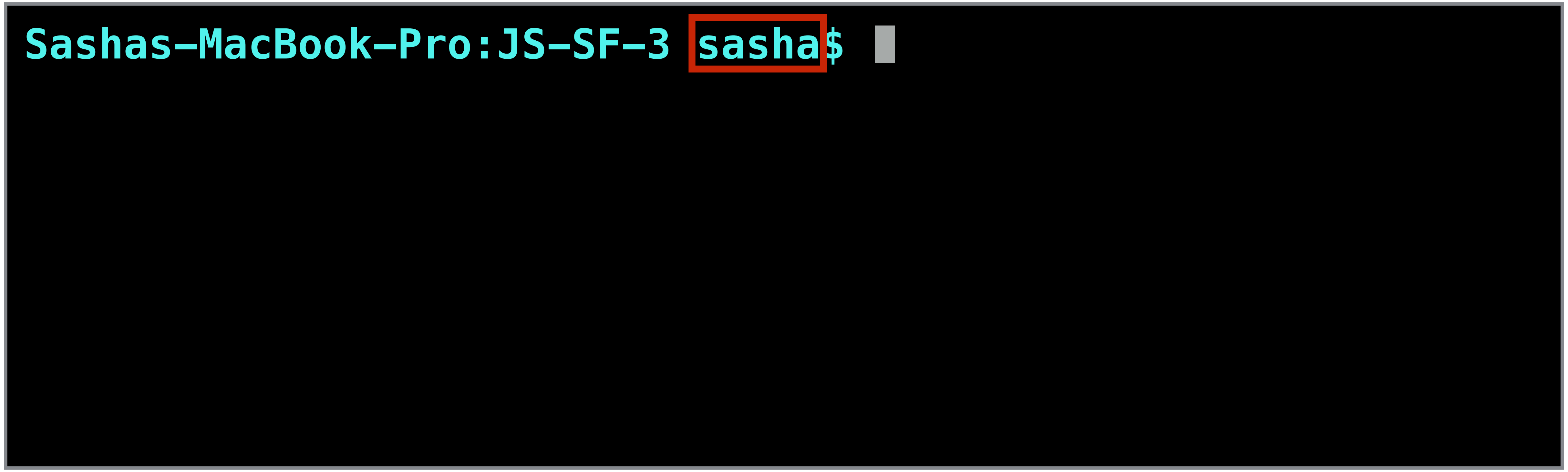
Working directory (current folder)



```
Sashas-MacBook-Pro:JS-SF-3 sasha$
```

ANATOMY OF THE TERMINAL

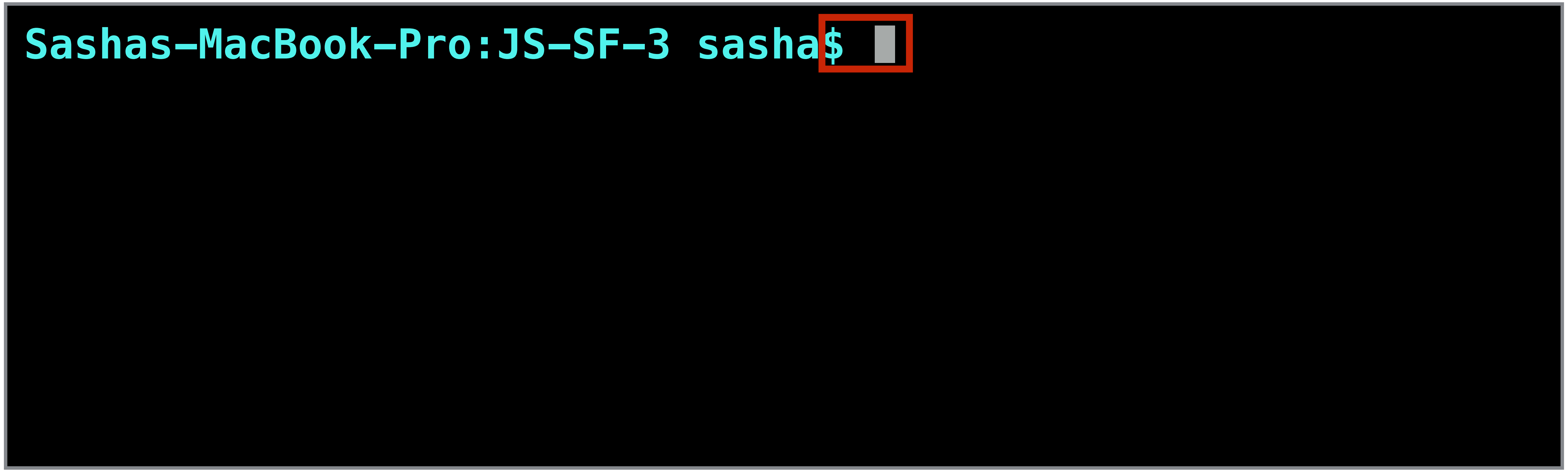
Username



```
Sashas-MacBook-Pro:JS-SF-3 sasha$
```

ANATOMY OF THE TERMINAL

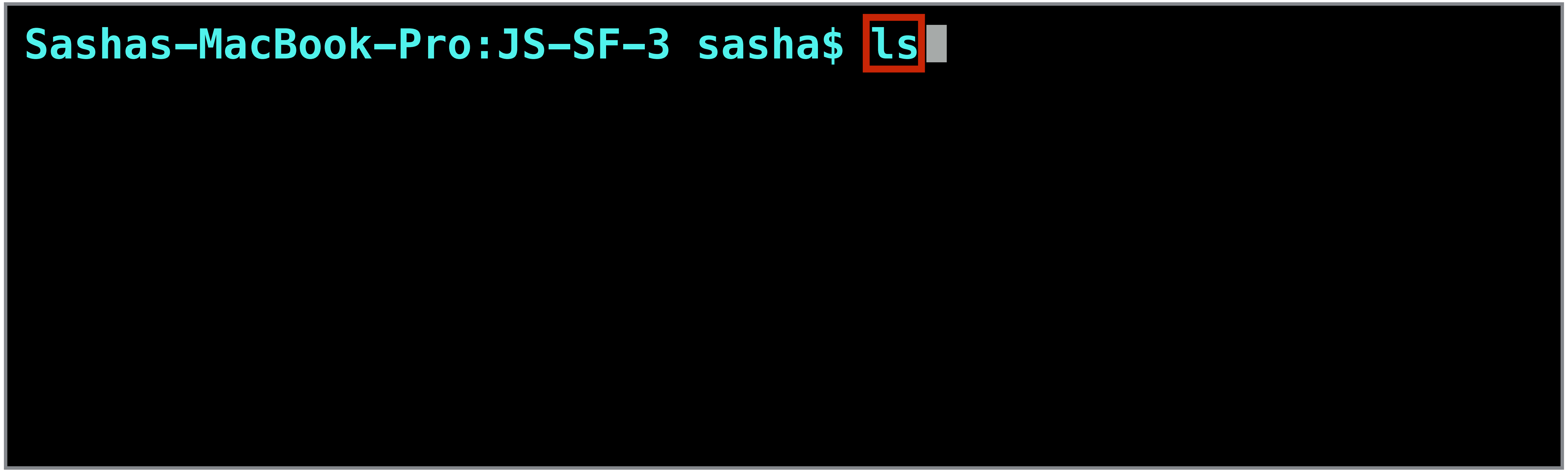
Bash prompt



```
Sashas-MacBook-Pro:JS-SF-3 sasha$ 
```

ANATOMY OF THE TERMINAL

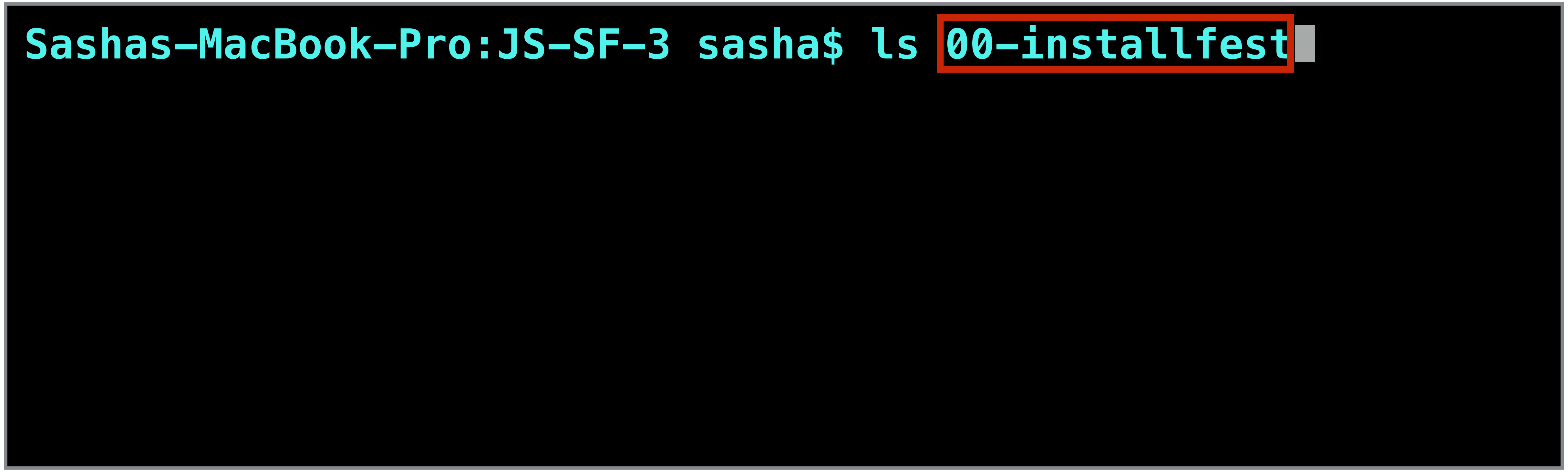
Command (program)



```
Sashas-MacBook-Pro:JS-SF-3 sasha$ ls
```

ANATOMY OF THE TERMINAL

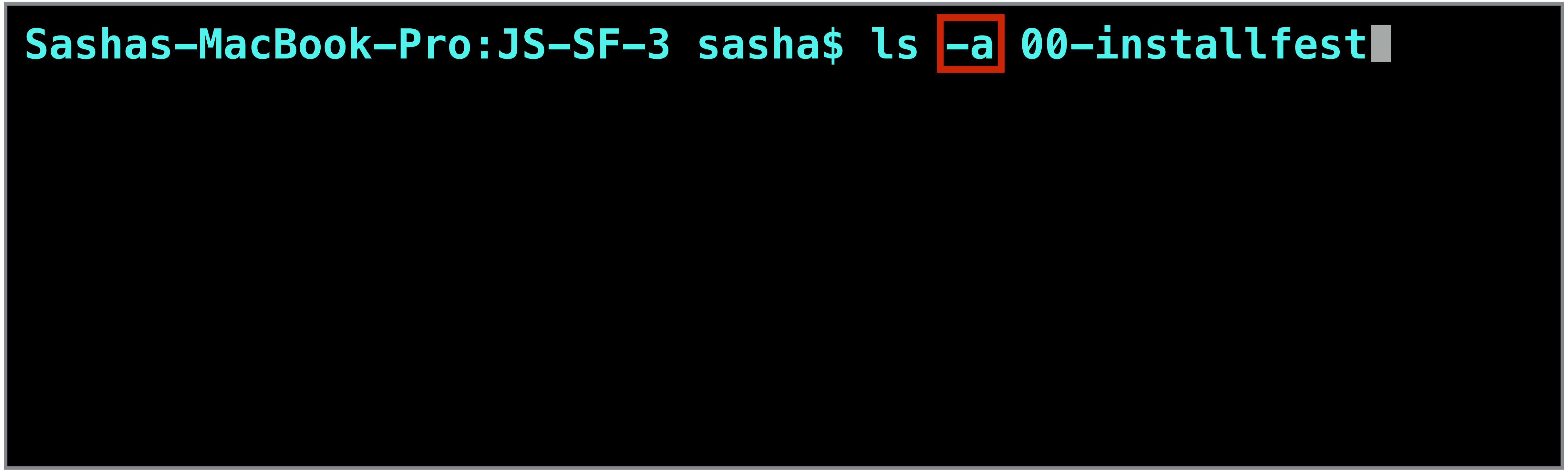
Argument (input)



```
Sashas-MacBook-Pro:JS-SF-3 sasha$ ls 00-installfest
```

ANATOMY OF THE TERMINAL

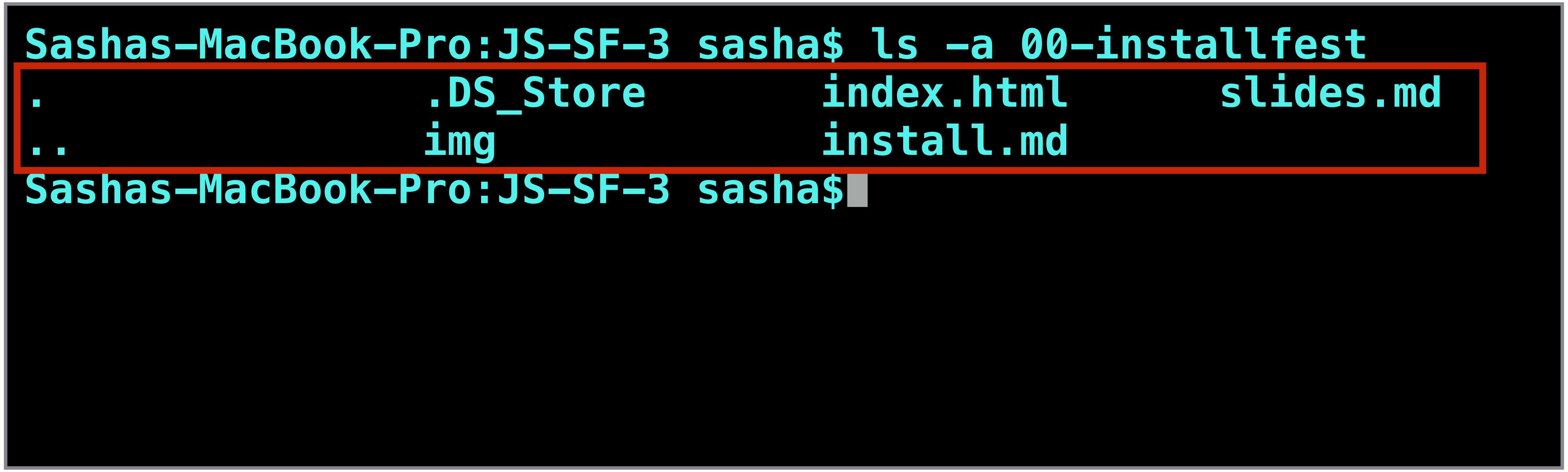
Option



```
Sashas-MacBook-Pro:JS-SF-3 sasha$ ls -a 00-installfest
```

ANATOMY OF THE TERMINAL

Output



```
Sashas-MacBook-Pro:JS-SF-3 sasha$ ls -a 00-installfest
.                .DS_Store        index.html       slides.md
..               img              install.md
Sashas-MacBook-Pro:JS-SF-3 sasha$
```

(UNIX) COMMAND LINE BASICS

Command	Description
`ls`	**L**i**s**t the contents of the directory
`cd`	**C**hange **d**irectories
`mkdir`	**M**a**k**e **dir**ectory
`rmdir`	**R**e**m**ove empty **dir**ectories
`rm`	**R**e**m**ove files or directories
`touch`	Create an empty file
`echo`	Return a string

(UNIX) COMMAND LINE BASICS (CONTINUED)

Command	Description
`code`	Open Microsoft Visual Studio Code
`pwd`	**P**rint **w**orking **d**irectory
`say`	Make your computer talk
`open`	Open a file in default application
`man`	Show the **man**ual for a command
`cat`	Show the contents of a file
`clear`	Clear the terminal

FOLDER NAVIGATION PATHS

Path	Description
`./`	current directory
`../`	up one directory
`~/`	home directory

FOLDER NAVIGATION COMMANDS

Command	Description
`cd myFolder`	go into `myFolder`
`cd ..`	go up one folder/directory
`cd ~/`	go to home folder/directory
`cd`	shorthand for `cd ~/`

BREAK (5 MINUTES)

Command line codealong

For Mac

Open the Terminal app (Applications > Utilities > Terminal)

For Windows

Open the Git BASH application

Configure Visual Studio Code so you can call it from the command line

For Mac

https://code.visualstudio.com/docs/setup/osx

For Windows

(no configuration required)

Command line codealong

Command line exercise on your own

Introduction to Git/GitHub

Git is a tool that

- Was developed in the late 70s by same group of developers who made bash
- Primarily stores code, but can also store files like Dropbox or Google Drive does
- Maintains each file’s history (like Apple's Time Machine software)
- Is now commonplace in any company that employs engineers

Why is Git so popular with developers?

- Because Git stores a history of the code, it allows developers to roll code back to an earlier point in time if something breaks
- Git facilitates collaboration, and prevents developers from stepping on one another's toes
- Git tracks changes so you can see who worked on what

What is GitHub?

GitHub is a web app/platform that

- Facilitates the sharing and managing of code, making it easy for multiple engineers to collaborate on the same project
- Hosts files on the web so you can share the finished product with other people

Why is GitHub so popular with developers?

- Much like Dropbox or Google Drive lets multiple people collaborate on the same document, GitHub allows this for code.
- GitHub allows team members to provide feedback on the code, which potentially increases code quality
- Has project management features built in: issue tracking, delegation, etc

Git vs GitHub

- **Git** is version control software
- **GitHub** is a website and platform for utilizing Git in a collaborative way

Git/GitHub Vocabulary

- Repository
- Clone
- Commit
- Push

GitHub – Repository/Repo

- Basic element of GitHub
- Contains all of a project's files (all the code)
- One or more users can contribute to a single repository
- Repositories are either public or private
- By the end of class today, you will create your own repo

GitHub – Clone

- Copies/clones a **remote** repo to your machine
- This copy/clone is called a **local** repo
- Changes to the **local** repo will not affect the **remote**

GitHub – Commit

- A snapshot of changes to a repo
- Think of it as "saving" your changes
- Contains a message describing the changes made

GitHub - Push

- "Pushes" your commits (saved changes) to a **remote** repository
- Allows other developers to see your changes and "pull them down" to their own local repos

GitHub – How will we use it?

- You will clone the class repository for local access to homework, documentation, and project files
- You will submit your homework by pushing it to the repo and making a pull request
- The instructional team will provide feedback by commenting on your pull requests

GIT COMMANDS

Command	Description
`git init`	Create new repository
`git status`	See the status of files in your repo
`git log`	Look at a list of commits
`git add`	Add files to the stage for committing
`git commit`	Create a snapshot of your project
`git push`	Push updates to GitHub
`git pull`	Pull updates from GitHub

BREAK (5 MINUTES)

GitHub exercise

Intro to Node.js and command line JS

The terminal

How is Node different from JS in the browser?

- No browser-specific functionality
- Same JS engine as Chrome

What is Node good for?

- Creating a backend server for a web application
- Running a script to do data analysis
- File management
- Making command line programs

Ways to run Node

- Interactive command line
- Run a file

Executing JavaScript code

Let's write some JavaScript!

Variables

- Containers that allow us to store values
- Let us tell our program to remember values for us to use later on
- The action of saving a value to a variable is called **assignment**

Variable declaration

- Statement saying that we wish to create a variable

Variable assignment

- Specifying the value we wish to assign to a variable

Variable assignment and declaration

- We can do both in a single statement

`console.log`

- Logging to the console is how we print things out for our own inspection

Inspecting variables

```
console.log(y)
```

When do you use console.log?

- When you are developing a program and need help figuring out what’s going on (aka debugging)
- When you want to print things to the command line

JS exercise

Exit the Node console

CTRL + c twice

REVIEW

What does GitHub do and why will we be using it?

What is Node and what did we use it for today?

Next class preview: Data Types

- Describe the concept of a "data type" and how it relates to variables.
- Declare, assign to, and manipulate data stored in a variable.
- Create arrays and access values in them.
- Iterate over and manipulate values in an array.

Exit Tickets!

Scheduling

- Snack rotation
- Happy Hour (GA buys the first round!)

See SFJS5 repo for

- Pre-reading (optional)
- Additional resources on today’s topics

Q&A