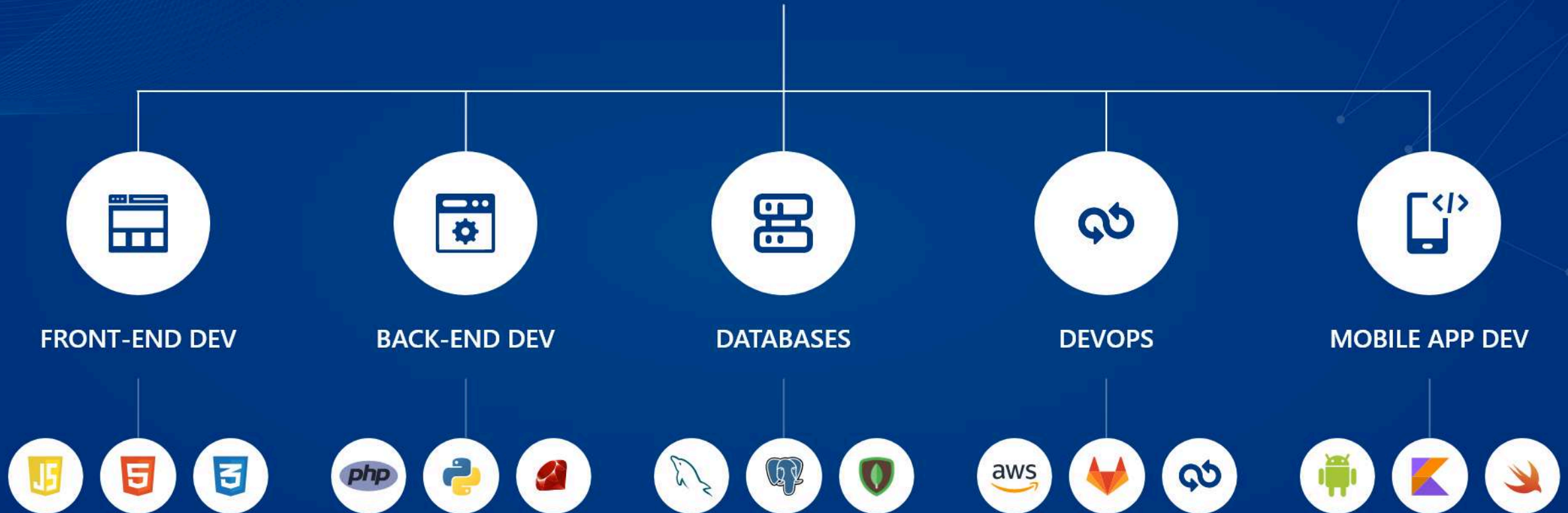


# Full-stack Web Development

**Aaron Zhang**

# Full Stack Web Development



## fullstack roadmap

### Front-end

- HTML + CSS + JS
- UI / UX
- Accessibility
- Browser
- DOM manipulation
- ...

### Back-end

- Java / Python / Node.js
- Database
- APIs
- Authentication
- Web security
- ...

Rank	Job Title	Mean Annual Salary (Above 2021 Nat'l Avg: \$58,260 or higher)	Jobs Per 1M, Period ending 2023-01-01	% change in job share, Period ending 2020-01-01 vs Period ending 2023-01-01	% containing remote + hybrid phrases, Period ending 2023-01-01
1	full stack developer	129,637	1,398	56	51
2	data engineer	135,260	879	80	44
3	cloud engineer	133,114	678	65	42
4	psychiatric nurse	109,739	537	45	18
5	senior product manager	147,135	532	44	45
6	back end developer	148,827	429	81	60
7	site reliability engineer	153,134	377	121	55
8	machine learning engineer	153,252	246	53	37
9	psychiatric mental health nurse practitioner	134,011	230	180	20
10	product designer	121,363	213	39	48

# Training Overview

- Instructors: Aaron Zhang & Alex Chen (Thursday TA Session)
- Content & Schedule
- Assignments & Quiz
  - Every Thursday - Quiz
  - Leetcode - **~150 problems solved before the end of the training**
- Projects
  - two group projects
- [PingCAP](#)

## Advanced Certifications



### **PingCAP Certified TiDB Associate Database Administrator**

Certify your knowledge of deploying, managing, and operating workloads on TiDB and TiDB Cloud.

[View Exam Details ↗](#)

### **PingCAP Certified TiDB SQL Developer**

Certify your ability to write and deploy applications on TiDB and TiDB Cloud.

[View Exam Details ↗](#)

### **PingCAP Certified TiDB Professional Database Administrator**

Certify your possession of the full skill set required to perform day to day administration of the TiDB and TiDB Cloud.

[View Exam Details ↗](#)

# Assessment & Grading

- 30% Quiz
- 35% Project 1
- 35% Project 2
  - **Live demo (advanced to marketing)**
  - Recording demo
  - Peer reviews

# How to succeed in this training

- Be curious
- Be proactive
- Be persistent
- Be collaborative
- **Be cautious about AI coding assistants**



# Preparation prior to courses

set up the development environment

- Browser: **Chrome** / Firefox / Safari
- Code editor / IDE: **Visual Studio Code**, WebStorm
- Git / GitHub
- Set up Node.js env (optional for now)
- ~~AI coding assistant: Copilot~~

# What is Javascript

- a programming language
- a scripting language
- a dynamic language
- a weakly typed language
- a prototype-based language
- a multi-paradigm language
- a single-threaded language
- ...

```
const pluckDeep = (key) => (obj) =>
  key.split('.').reduce((accum, key) => accum[key], obj)

const compose =
  (...fns) =>
  (res) =>
    fns.reduce((accum, next) => next(accum), res)

const unfold = (f, seed) => {
  const go = (f, seed, acc) => {
    const res = f(seed)
    return res ? go(f, res[1], acc.concat([res[0]])) :
    acc
  }
  return go(f, seed, [])
}
```

# Client-side Javascript vs Server-side Javascript

- Client-side Javascript
  - run in browser
  - manipulate DOM
  - make HTTP requests
  - ...
- Server-side Javascript
  - run in Node.js
  - manipulate files
  - make HTTP requests
  - ...

# History of Javascript

## ECMAScript

- 1995: Brendan Eich, Netscape
- 1996: ECMAScript 1
- 1997: ECMAScript 2
- 1999: ECMAScript 3
- 2009: ECMAScript 5
- **2015: ECMAScript 6**
- 2016: ECMAScript 7
- ...

# First Javascript Program

```
console.log('Hello World');
```

- run in browser console
- run in Node.js
- run in VS Code

# Lexical Structure

- case sensitive
- whitespace
- Unicode

```
'café' === 'caf\u00e9' ;
```

- comments

- identifiers
  - an identifier == a name
  - start with a letter, underscore `_`, or dollar sign `$`
- reserved words
  - `break`, `case`, `catch`, `class`,  
`const`, `continue`, `debugger`,  
...

# Lexical Structure (cont'd)

- literals - a data value that appears directly in a program
  - 123, 12.34, 'abc', "abc", true, false, null, undefined, /abc/, ...
- optional semicolons
  - `;` is optional, but it is required when multiple statements appear in a single line
  - In general, if a statement begins with `(`, `[`, `/`, `+`, or `-`, it could be interpreted as a continuation of the statement before.

```
var x = 0; // Semicolon omitted here
[x, x + 1, x + 2].forEach(console.log);
```

# Data Types

## Primitive Types

- **number**
- **string**
- **boolean**
- **null**
- **undefined**
- **symbol**
- *bigint (ES2020)*

## object / complex

- **Object**
- **Array**
- **Function**
- **Date**
- **RegExp**
- ...



# Number

- integer
- floating-point
- NaN
- Infinity / -Infinity => Number.POSITIVE\_INFINITY /  
Number.NEGATIVE\_INFINITY
- Number.MAX\_VALUE / Number.MIN\_VALUE

# String

- single quote `'` / double quote `"` (*no character type in Javascript*)
- backtick quote ```
- escape character `\`
- string length - maximum length is  $2^{53} - 1$ , but it depends on the browser
  - In V8:  $2^{29} - 24$  (~1GiB),  $2^{28} - 16$  (~512MiB) on 32-bit system.
  - In Firefox,  $2^{30} - 2$  (~2GiB). Before Firefox 65, the maximum length was  $2^{28} - 1$  (~512MiB).
  - In Safari,  $2^{31} - 1$  (~4GiB).

# Boolean

- `true` / `false`
- `Boolean()` function

# Null & Undefined

- `null` - a special value that indicates a deliberate non-value
- `undefined` - a special value that indicates an uninitialized value

# Symbol

- a new primitive type in ES6
- a unique and immutable data type
- used as the key of an object property

could be used to replace constants

```
const COLOR_RED = Symbol('red');  
const COLOR_GREEN = Symbol('green');  
const COLOR_BLUE = Symbol('blue');
```

# Type Conversion / Coercion

- difference between conversion and coercion: coercion is implicit, conversion is either implicit or explicit
- `Number()` function
- `String()` function
- `Boolean()` function
- `parseInt()` / `parseFloat` function

# Dynamic Typing

- a variable can hold a value of any type
- a variable can hold different types of values at different times
- a variable can be reassigned to a value of a different type

# Variable Declaration

- `var` keyword
- `let` keyword
- `const` keyword
- naming convention
  - camelCase
  - snake\_case
  - PascalCase



# Variable Scope

- global scope
- function scope
- block scope (ES6)

## Hoisting

```
console.log(x); // undefined  
var x = 1;
```

# Operators

- arithmetic operators
- assignment operators
- comparison operators
- logical operators
- string operators
- conditional (ternary) operator
- unary operators
- bitwise operators
- comma operator
- `typeof` operator
- `delete` operator
- `in` operator
- `instanceof` operator
- `void` operator